UNIVERSITÀ DI PARMA
DIPARTIMENTO DI INGEGNERIA E ARCHITETTURA

UNIVERSITÀ DEGLI STUDI DI PARMA

Dottorato di Ricerca in Tecnologie dell'Informazione
XXXVII Ciclo

Mattia Pellegrino

Artificial Intelligence in Healthcare: From Agents Simulation to Cyber-Physical Systems

DISSERTAZIONE PRESENTATA PER IL CONSEGUIMENTO
DEL TITOLO DI DOTTORE DI RICERCA

ANNI ACCADEMICI 2021/2022 – 2023/2024

UNIVERSITÀ DI PARMA

DIPARTIMENTO DI INGEGNERIA E ARCHITETTURA

UNIVERSITÀ DEGLI STUDI DI PARMA

Dottorato di Ricerca in Tecnologie dell'Informazione

XXXVII Ciclo

Artificial Intelligence in Healthcare: From Agents Simulation to Cyber-Physical Systems

Coordinatore:

Chiar.mo Prof. Marco Locatelli

Tutor:

Chiar.mo Prof. Agostino Poggi

Co-Advisor:

Chiar.mo Prof. Gianfranco Lombardo

Dottorando: *Mattia Pellegrino*

Anni Accademici 2021/2022 – 2023/2024

"There is no doubt that it is around the family
and the home that all the greatest virtues
are created, strengthened, and maintained."
Winston Churchill

Abstract

The integration of Artificial Intelligence (AI) in healthcare is rapidly reshaping this scenario, offering unparalleled opportunities to enhance patient care, improve the process of medicine, and enable predictive analytics. Recent developments in AI, mainly through Agent-Based Modeling and Simulation (ABMS) and Cyber-Physical Systems (CPSs), provide innovative tools that can consider the complexity and scale of modern challenges faced in healthcare. From diagnostics to treatment planning and real-time monitoring to predictive modeling, AI-driven systems are helping to bring much-needed personalization, efficiency, and data-informed decision-making into medical settings. With the increase in healthcare data volume and complexity, advanced AI methodologies like ABMS and CPS become instrumental in properly managing and interpreting such information.

ABMS offers a unique capability in simulating complex, agent-driven environments where individual agents, patients, medical professionals, or healthcare resources interact within a digital framework. This is useful in epidemic modeling to forecast the spread of diseases and assess various intervention strategies. ABMS has proved instrumental in providing health response intelligence by letting researchers test how variations in social behaviors and policies might have an impact, most recently in the COVID-19 pandemic.

On the other hand, CPSs use real-time data integration and AI algorithms to develop systems that achieve dynamic monitoring and control in medical settings. Moreover, CPSs can integrate Internet of Things (IoT) devices and Digital Twin (DT) technology to realize a continuous data flow between the physical and digital environ-

ments. Therefore, health professionals can safely and efficiently monitor patient data and optimize processes during their activities. Digital Twins, which mimic physical environments in real-time, further enable CPS applications to perform simulations that can effectively enhance resource allocation and patient care in hospital settings.

However, implementing AI in healthcare also has some significant challenges. Some of the problems with its effectiveness include the shortage of access to diverse and complete medical data, compatibility with legacy infrastructure, and biases within AI algorithms. Similarly, data availability will be restricted because of privacy issues and strict regulations. Next, integrating AI into healthcare facilities could become rather complex due to outdated systems. If not validated with care, AI models may even perpetuate already existing biases related to race, gender, or socioeconomic status, which would impact patient outcomes. Those bottlenecks are what ABMS and CPS can address with very strong strategies in the improvement of data-driven healthcare processes. They allow for simulations and real-time monitoring, making it easy for healthcare professionals to make informed decisions that improve operational efficiency while ensuring better patient outcomes.

This research underscores AI's pivotal role in building more efficient, intelligent healthcare systems and addressing critical challenges along the way. The work carefully discusses the two methodologies, ABMS and CPS. First, it looks into ABMS to analyze complex agent-driven interactions that could be simulated for better planning, resource management strategies, or even a response to complex health crises. In addition, it talks about CPS and how real-time integration of data with digital twin technology will lead to enhanced patient monitoring and operational control in a clinical setting. These approaches provide a solid foundation to meet the challenges regarding data accessibility and compatibility with legacy systems. In this dual analysis, the work can provide meaningful learning for how AI can be used effectively and responsibly to bring more adaptability and equity into the healthcare landscape.

Contents

List of Figures

List of Tables

Introduction

The rise of Artificial Intelligence (AI) is revolutionizing several industries, one of the sectors that is gaining the most from this technology is healthcare. As the healthcare system becomes more complex, so does the volume and variety of data generated, requiring advanced tools to efficiently leverage this information. AI has been a part of healthcare since the 1950s when the first attempts were made to improve diagnoses using computer-aided programs [1, 2]. Medical AI applications have become popular lately because of the way modern computers have become much more powerful and the huge amount of digital data that's out there for us to collect and use [3]. AI's ability to process huge amounts of data and recognize patterns has the potential to revolutionize patient care by improving diagnostic precision and streamlining treatment methods. AI can be integrated into numerous aspects of medicine, including diagnostics, clinical care, rehabilitation, surgery, and predictive analytics. A significant contribution of AI in healthcare is in aiding clinical decision-making and diagnosing diseases. By analyzing extensive datasets from various sources, AI can identify illnesses and assist healthcare professionals in making informed clinical choices [4]. Moreover, AI-driven simulations allow healthcare providers to model complex medical scenarios and predict outcomes in a controlled environment. These simulations can be used for training, planning surgical procedures, and testing treatment plans, finally enhancing the precision and safety of medical interventions. By incorporating AI in simulation, healthcare professionals can explore a wide range of potential scenarios, improving their preparation and the overall quality of patient care.

The historical development of artificial intelligence in the health sector provides a

foundation for modern research aimed at solving new and upcoming problems. While early efforts focused mainly on diagnostic systems, advances in computational capability and data availability enabled the development of more sophisticated applications: predictive modeling, tailored therapeutics, and autonomous surgical systems. These advances underline the potential of artificial intelligence in driving the development of more responsive and adaptive healthcare systems. This contextual setting calls for more specific research in artificial intelligence technologies to address unmet healthcare needs.

This research builds upon many years of progress and discusses how AI is being used in healthcare. It highlights new developments in using simulation technologies and cyber-physical systems. This thesis explores how AI can be used in healthcare, with a particular view toward CPS and ABMS as the main tools for solving some of the biggest challenges in this field.

Main challenges

Despite its promise, AI implementation in healthcare has to face several challenges. One of the main issues is the restricted access to comprehensive and various healthcare data. Privacy concerns, severe regulations, and the sensitive nature of medical data often restrict the availability of high-quality datasets that are essential for training robust AI models. This limitation also affects the development of accurate predictive models and raises questions about the generalizability of these models across different patient populations.

Meeting such challenges will require new data-sharing frameworks that balance privacy utility and collaboration in building diverse, high-quality datasets. The integration of AI with existing legacy healthcare infrastructures represents another significant challenge. Healthcare systems are often built on legacy platforms incompatible with AI technologies, requiring important upgrades and adaptation to coordinate everything. Moreover, AI systems should work effectively even across various healthcare settings, from urban hospitals to rural clinics, without compromising performance or accuracy.

Additionally, bias in AI algorithms is another major problem. AI models, if not

carefully designed and validated, can make existing biases in healthcare even worse, especially those related to race, gender, and socioeconomic status. This issue is critical because biased decision-making can seriously affect patient outcomes.

Finally, equipment installation and utilization in clinical environments create a range of important electromagnetic and regulatory challenges. These include controlling electromagnetic field interference that can render sensitive medical devices inoperable, maintaining asepsis, and capturing high-quality real-world data essential to artificial intelligence development and implementation. The above challenges, in addition to strict regulatory requirements, are essential to guarantee the safety, reliability, and effectiveness of artificial intelligence technologies in clinical environments.

Overview of the Thesis

This thesis is structured around two key AI methodologies: CPS and ABMS. These methodologies have the potential to address challenges in AI-driven healthcare.

- **ABMS in Epidemic Simulation**

  This part delves into the use of ABMS as a powerful tool for epidemic simulation and healthcare scenario planning. ABMS supports a multi-agent system where individual agents, such as patients, healthcare workers, and other entities, interact in a simulated environment. This particular method enables researchers to model complex healthcare scenarios, such as the spread of an infectious disease, and predict the outcomes of various intervention strategies. Moreover, ABMS is particularly valuable in situations where real-world data is limited, incomplete, or difficult to obtain, such as during the early stages of an epidemic or in under-researched areas of healthcare. With ABMS, researchers can generate synthetic data that compensates for these gaps. By simulating various scenarios and generating data that mirrors potential real-world conditions, ABMS provides crucial insights for resource allocation, emergency response planning, and the optimization of healthcare processes, especially during unpredictable and dynamic events like epidemics.

- **CPS in Healthcare**

  As referred to in this thesis, the CPS involves integrating devices and AI algorithms to form a unified system capable of collecting, processing, and analyzing data in real time. This facilitates monitoring and managing patient health in dynamic environments such as hospitals and surgical theaters.

  For instance, sensors can be strategically placed within the CPS framework to continuously gather patient data, including vital signs, surgical progress, and other critical health metrics. AI algorithms process and analyze this data, identifying patterns, predicting outcomes, and generating actionable insights for healthcare providers.

  A key aspect of CPS is its ability to create digital replicas, or digital twins (DT), of physical environments. These virtual models reflect real-time conditions in healthcare settings, allowing for precise, data-driven decision-making, improved surgical outcomes, and enhanced patient safety. CPS represents a versatile approach to advancing healthcare by integrating AI technologies.

In my thesis work, I aim to advance the understanding of AI's potential in healthcare by providing a detailed analysis of how the CPS and ABMS can be used to address key challenges in this particular field. By exploring these methodologies, I offer new insights into the practical applications of AI for developing more effective, efficient healthcare systems. My research aims to contribute to developing more intelligent, data-driven healthcare environments that can better aid patient populations and improve overall health outcomes.

To summarize, the specific objectives of this study are to:

- Discuss the application of CPS in real-time health monitoring and decision-making.
- Analyze how ABMS can enhance epidemic modeling and resource allocation.
- Propose strategies for overcoming challenges in integrating AI technologies in a real scenario with physical constraints and heterogeneous existing devices.

Chapter 1

State of the art

Integrating AI in the healthcare context can revolutionize medical practices, from diagnosis to treatment planning. However, this integration is full of challenges, such as privacy concerns, integration with legacy infrastructure, and the need to generalize across different batches of population. In this section I will report on the current state of AI in healthcare, focusing on two critical methodologies that are the subjects of this thesis: ABMS and CPSs.

ABMS has come up to be a powerful tool in healthcare, enabling the detailed modeling of complex systems by simulating the interactions between individuals represented as agents, such as people, patients, or healthcare providers. ABMS allows the simulation of large-scale, high-resolution scenarios, unlike the traditional modeling techniques, capturing the complex behaviors and their impact on the simulated environment. Additionally, ABMS can generate synthetic data mimicking real-world complexities, useful for training AI models when real data is insufficient or sensitive. It simulates diverse scenarios to produce datasets reflecting various patient behaviors and treatment responses. This ability helps in developing robust, widely applicable AI-driven predictive models, allowing testing and validation across simulated conditions. This approach has become increasingly relevant in scenarios requiring a granular understanding of interactions within healthcare systems.

On the other hand, CPSs in healthcare are gaining attention as they integrate phys-

ical systems with digital technologies, enabling real-time monitoring, simulation, and decision-making. These approaches leverage AI in combination with digital tools to create systems that can link the physical to the digital world. For instance, the development of DTs within cyber-physical systems has opened new ways to optimize healthcare scenarios, improving patient outcomes and enhancing medical accuracy. The integration of Machine learning (ML) and predictive analytics into these frameworks remains poorly explored, presenting challenges and opportunities for future research.

In this section, I will explore these methodologies and the challenges they provide, providing a foundation for the research and the innovation presented in this thesis.

1.1 Epidemic Simulation

Epidemic simulation plays a pivotal role in understanding and managing the spread of an infectious disease. The COVID-19 pandemic highlighted how much the global community is vulnerable and the critical need for advanced modeling techniques to predict and adopt strategies to control the outbreaks.

Traditional methods of epidemic modeling provided optimal valuable insights into simulating the dynamics and the potential of strategies in disease simulations. However, modern pandemics require more complex and detailed approaches that can account for different variables and individual behaviors. Such needs make AI increasingly important, offering new possibilities for enhancing epidemic simulations' accuracy and effectiveness.

AI-driven tools allow for integrating huge amounts of data, improving the simulation's granularity and increasing the predictions' accuracy. AI is particularly useful in scenarios when real-time decision-making is required, such as allocating resources or implementing measures to mitigate the disease spread. Using such tools in epidemic simulation can help researchers understand the current spread pattern and prepare us for new outbreaks, refining the ability to anticipate and respond in the right way to any emerging threats.

This section will explore the state of the art in epidemic simulation, focusing on two key methodologies: System Dynamics (SD) and ABMS.

1.2 System Dynamics

SD is a powerful approach to understanding and managing complex systems. It was originally developed in the mid-1950s by Professor Jay W. Forrester to help corporate managers improve their understanding of industrial processes [5]. It is widely used to analyze a range of systems, such as business, ecology, and medicine. Particularly, when applied to epidemics, SD offers a way to model and simulate a disease spread within a population.

SD results from creating models of systems. These models set limits on how operations can be optimized to ensure fairness. When dealing with a specific problem, the system's dynamics must be based on the underlying physics. This can often be shown using differential equations:

$$\dot{x} = f(x,u) \tag{1.1}$$

where $f(x,u)$ represents the SD in terms of a state variable x and the manipulated variable u. Other dynamical processes are the simple modelling of an integrator of the following discrete form:

$$x(k+1) = x(k) + u(k) \tag{1.2}$$

In many real-world scenarios, the components of a system are not isolated but are interconnected, forming complex networks. These networks can represent various relationships, such as communication links in a social network, interactions between species in an ecosystem, or even the connections between different departments within an organization.

When applying SD to such networks, the focus shifts from analyzing individual elements to understanding how the structure and interconnections within the network influence the overall system behavior. Each node in the network can represent a system state, while the edges denote the interactions or flows between these states. The

dynamics of the entire network can then be captured by a set of coupled differential equations, where the state of each node is influenced not only by its own dynamics but also by the states of its neighboring nodes.

For instance, in the context of disease spread, each node could represent an individual or a group of individuals, and the edges could represent the potential pathways for disease transmission. The dynamics of the disease spread across the network could be modeled as:

$$\dot{x_i} = f(x_i, u_i) + \sum_{j \in N(i)} g(x_j, u_j) \tag{1.3}$$

Here, $\dot{x_i}$ represents the rate of change of the state at node i, which depends on the internal dynamics $f(x_i, u_i)$ and the influence of neighboring nodes $N(i)$ through the interaction term $\sum_{j \in N(i)} g(x_j, u_j)$. This approach allows for the study of how localized changes can propagate through the network, leading to emergent behavior that cannot be easily predicted by examining individual components in isolation.

Therefore, SD on a network provides a powerful framework for analyzing how complex systems evolve over time, considering both the internal processes of individual elements and their interactions within a broader interconnected structure. This perspective is crucial in fields like epidemiology, where the spread of disease is heavily influenced by the network of contacts within a population, or in organizational management, where the flow of information and resources between departments can determine the success or failure of strategic initiatives.

1.2.1 SEIR Model

When system dynamics are integrated into epidemic modeling, the resulting main models include the SEIR model. It describes how individuals flow in the course of an infection by parameters of transmission, incubation, and recovery rates. This model should be useful as it can handle the simulation of disease spread and the evaluation of intervention strategies over time. The SEIR model was developed in the early 20th century, with the most notable work by Kermack and McKendrick [6]. In SD, the use of compartmental models allows us to break down complex systems into

manageable, interconnected parts. This provides the opportunity for a more detailed analysis concerning the flows and transitions between various states of the system and offers deeper insights into how the various drivers influence behavior as a whole. Moreover one of the most successful implementations of compartment modeling is in infectious diseases studies where the population is divided into different health states.

The SEIR model is an extension of the basic SIR (Susceptible-Infected-Recovered) model, which includes an additional compartment for "Exposed" individuals who have been infected but are not yet infectious. The SEIR model is a common approach in epidemiology to simulate the spread of infectious diseases. The system consists of four compartments:

- **S (Susceptible)**: Individuals who are healthy but have not yet contracted the disease.
- **E (Exposed)**: Individuals who have been infected with the pathogen but are not yet infectious. These individuals are in the incubation phase of the disease.
- **I (Infectious)**: Individuals who have contracted the disease and can transmit it to susceptible individuals.
- **R (Recovered)**: Individuals who have recovered from the disease are assumed to be immune, meaning they cannot be re-infected.

Figure 1.1 shows the flow chart of the SEIR model.

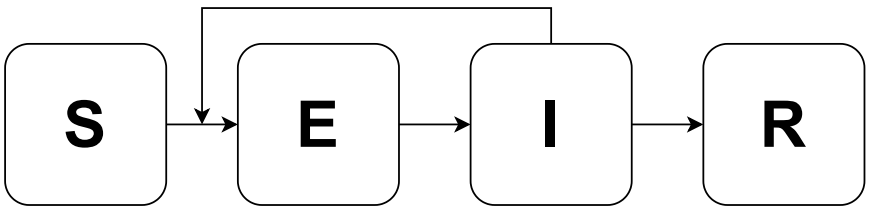

Figure 1.1: Flow diagram of SEIR model

In the SEIR model, individuals move from one compartment to another based on the progression of the disease. The transitions between these compartments are

governed by several parameters:

- β: The transmission rate, which defines how frequently a susceptible individual comes into contact with an infectious individual and becomes exposed.

- σ: The rate at which exposed individuals become infectious. The inverse of this rate, $\frac{1}{\sigma}$, is the average incubation period.

- γ: The recovery rate of infectious individuals, which defines the average duration of the infectious period. The inverse of this rate, $\frac{1}{\gamma}$, gives the mean time that an individual remains infectious.

- **Susceptible** (S): The initial phase where individuals have no immunity and are susceptible to contracting the disease. Individuals in this compartment become exposed (E) when they come into contact with infectious individuals (I) at a rate proportional to β. The number of new exposures per unit time is βSI.

- **Exposed** (E): Individuals move from the exposed phase to the infectious phase at a rate σ. The rate σ corresponds to the inverse of the average incubation period. Therefore, the number of individuals leaving the exposed compartment per unit time is σE.

- **Infectious** (I): Once an individual becomes infectious, they stay in this compartment for an average time of $\frac{1}{\gamma}$. During this time, they can transmit the disease to susceptible individuals. The rate at which they recover and move to the recovered compartment is given by γI.

- **Recovered** (R): Individuals in the recovered compartment are assumed to have immunity to the disease, so they no longer participate in the dynamics of the disease spread.

The transitions between these compartments can be modeled by a system of ordinary differential equations (ODEs). Let $S(t)$, $E(t)$, $I(t)$, and $R(t)$ represent the number of individuals in each compartment at time t. The equations are:

$$\begin{aligned} \frac{dS}{dt} &= -\beta SI \\ \frac{dE}{dt} &= \beta SI - \sigma E \\ \frac{dI}{dt} &= \sigma E - \gamma I \\ \frac{dR}{dt} &= \gamma I \end{aligned} \tag{1.4}$$

In addition to the standard SIR model, this variation assumes that the total population is conserved, i.e., $S+E+I+R=1$. As a result, the last differential equation becomes redundant. The initial conditions for the system are $S(0)>0$, $I(0)\geq 0$, $E(0)\geq 0$, and $R(0)=0$. For the infection to spread, the sum $E(0)+I(0)$ must be positive.

The main outcome of this model variation is that including a latent period can slow down the system's dynamics, without significantly reducing the overall spread of the disease. However, in the SEIR model, the growth rate after the pathogen's introduction is slower, as the infection must pass through the exposed class (E) before individuals become infectious.

Demographic factors become relevant if the focus is on the long-term persistence and dynamics of an infectious disease or on the spread of a scientific idea. A key factor in these scenarios is the influx of new susceptible individuals into the population, such as through births of individuals who have not been exposed to the disease.

In epidemiology, one common way to account for demographics in the SEIR model is to introduce a natural host lifespan, typically denoted as $\frac{1}{\mu}$ years. Here, μ represents the natural mortality rate, independent of the disease and unrelated to the pathogen's virulence. It is also usually assumed that μ represents the birth rate, keeping the total population size constant over time. This gives us the relation:

$$\frac{dS}{dt}+\frac{dI}{dt}+\frac{dR}{dt}=0 \tag{1.5}$$

which ensures population stability.

- **Susceptible Compartment** (S):

$$\frac{dS}{dt} = \mu(1-S) - \beta SI \tag{1.6}$$

This equation now includes a birth term, $\mu(1-S)$, which represents the influx of new susceptible individuals into the population, countering the natural mortality rate. The infection term βSI represents the rate at which susceptible individuals become exposed to the disease through contact with infectious individuals. As the susceptible population becomes infected, its size decreases.

- **Exposed Compartment** (E):

$$\frac{dE}{dt} = \beta SI - \sigma E - \mu E \tag{1.7}$$

This equation describes the rate of change in the exposed population, taking into account both the transmission of the disease and the natural mortality. The first term, βSI, represents the rate at which susceptible individuals become exposed. The second term, σE, accounts for the rate at which exposed individuals transition to the infectious stage. The new term $-\mu E$ accounts for the natural death rate in the exposed compartment.

- **Infectious Compartment** (I):

$$\frac{dI}{dt} = \sigma E - \gamma I - \mu I \tag{1.8}$$

This equation models the dynamics of the infectious population. The first term, σE, represents the rate at which exposed individuals become infectious. The second term, γI, accounts for the rate at which infectious individuals recover. The new term $-\mu I$ models the natural death rate among the infectious population.

- **Recovered Compartment** (R):

$$\frac{dR}{dt} = \gamma I - \mu R \tag{1.9}$$

This equation describes the rate of change in the recovered population. The term γI represents the rate at which infectious individuals recover and gain immunity. The new term $-\mu R$ accounts for the natural death rate in the recovered population.

Population conservation: In the modified SEIR model with demographics, the total population size remains constant over time, meaning that the sum of individuals in all compartments stays at 1:

$$S(t) + E(t) + I(t) + R(t) = 1$$

This is ensured by balancing the birth and death rates through the inclusion of μ in each equation.

Basic Reproduction Number (R_0): The basic reproduction number, R_0, remains a crucial metric for determining whether an infection will spread or die out. It is defined by:

$$R_0 = \frac{\beta}{\gamma + \mu}$$

This version of R_0 reflects both the recovery rate γ and the natural death rate μ, as both factors influence the duration of infectiousness.

Initial conditions: As before, the initial conditions specify the starting sizes of each compartment. Typically, at the start of an epidemic, the population consists mostly of susceptible individuals, with a small number in the exposed or infectious compartments. The recovered compartment is often empty at the outset. This can be expressed mathematically as:

$$S(0) = S_0, \quad E(0) = E_0, \quad I(0) = I_0, \quad R(0) = 0$$

By introducing the natural mortality and birth rates via μ, the SEIR model is better suited to studying the long-term dynamics of disease spread, particularly in

populations where births and deaths play a significant role. This model variant allows us to investigate how demographic factors affect the persistence of infection over extended periods and provides a more realistic framework for evaluating public health strategies in the context of both short-term outbreaks and long-term endemic diseases.

1.3 Agent-Based Modeling and Simulation

ABMS is a computational approach used to simulate complex phenomena to understand their behavior and underlying mechanisms [7]. This approach leverages multi-agent systems and simulation models [8]. In ABMS, simulations are built using a bottom-up, generative process that incorporates three key elements:

- Agents: which can have several behaviors.
- Environment: in which agents operate according to their internal state.
- Mechanism: that manages the interactions between agents, facilitating both direct and indirect information exchange [9].

The generative property of ABMS makes it especially useful for studying phenomena that are complex and difficult to set up with traditional simulation methods because it allows for analysis of both the whole system and its individual components [10].

ABMS is applied across numerous research and application domains, and the models can be different according to their inner characteristics that depend on the complexity of the phenomena being simulated. Common ABMS models used for complex systems include cellular automata [11], flocking systems [12], and geographical information systems [13]. One reason for the widespread adoption of ABMS is the availability of software platforms that simplify model development, simulation execution, and result analysis. Well-known platforms include ASCAPE [14], MASON [15], Repast [16], AnyLogic [17], and NetLogo [18]. However, these platforms often lack the tools needed to create fully intelligent agents, which may require integration with AI languages like Prolog or ML libraries. For instance, Chumachenko et al. [19] demonstrated a method for integrating declarative languages, showing how this can improve decision-making inference efficiency in simulation environments.

One reason that has facilitated the growth of ABMS applications is the development of software platforms to run and handle large-scale and complex scenarios. This has been further supported by rapid progress in technologies such as computer clustering and GPUs. Nevertheless, it remains challenging to develop large-scale ABMS applications involving many agents [20]. One problem is that agent actions drive simulations, and since there is a need for continuous interaction with other agents, the computational cost increases manifold. This poses problems in simulations exploiting computer clusters where efficiency is achieved based on processors performing computation tasks rather than communicating. Additionally, establishing result consistency through the maintenance of agent execution order needs synchronization protocols, which further slows down execution [21]. Finally, designing agents can be complex if heterogeneous agents need to be developed or if agents are to be configured with different or random data.

1.3.1 Applications

When ABMS is applied to social processes, the agents represent individual people or groups of people. All other agents in this case simulate the social interactions or relationships that any single given agent might have. The fundamental assumption lying behind such modeling is that it is possible to model people and their social interactions at a reasonable level of abstraction.

Thomas Schelling was the first to develop an agent-based social model. In that model, agents were interpreted as persons, and the interactions between agents expressed relevant social processes. Schelling did apply the notions of cellular automata to study models of residential segregation. The studies showed that ghettos could spontaneously form. More generally, Schelling showed how patterns may emerge that could not necessarily be implicit or even consistent with agents' objectives. This observation shifted the attention to the ABMS field. Agent-based modeling made it possible to simulate artificial societies, as was accomplished by Epstein and Axtell in 1996 [22].

Agent-based modeling can be used in multiple fields, as summarized in Table 1.1.

Industries and Businesses • Manufacturing • Markets • Assembly Lines • Insurance Economics • Artificial Financial Markets • Trade Networks Infrastructures • Electricity Markets • Hydrogen Economy • Transportation Population • Mass Movements • Evacuation Models	Culture and Society • Ancient Civilizations • Civil Disobedience Terrorism • Social Determinants • Organizational Networks Military • Command & Control Biology • Ecology • Animal Group Behavior • Cellular Behavior • Sub-molecular Models

Table 1.1: Applications that can be modeled using an agent-based system.

1.3.2 Building an ABMS System

The elaboration of an ABM does not differ in its very foundations from that of any other model or simulation. First, it identifies for what purpose the model will be used and what question the model should answer. Then, the system to be analyzed should be examined step by step: components are identified, interaction among them, relevant data, and so forth. Such steps are also used when developing an agent-based model.

An agent-based model has some necessities for being implemented correctly:

- Identify the agents and program their behavior;
- Identify relationships between agents and construct a theoretical model;
- Choose which ABMS platform to use and which modeling strategy to adopt;
- Obtain the data necessary to initialize the agents;

- Validate the models of agent behavior;
- Run the model and analyze the output from both a micro-simulation and macro-simulation perspective.

As mentioned earlier, agent identification, programming of their behavior, and representation interactively within realistic accuracy are a recipe for developing a functional agent-based model.

After the agents' definition, one needs to define what their main behaviors are. Once these are designed, either *knowledge engineering* or *participatory simulation* can be chosen to implement a hypothetic or existing system. *Knowledge engineering* is defined as a group of techniques for eliciting and structuring knowledge, given some reporting errors and situational biases. Information about agent behaviors is gathered using structured interviews.

ABMS participatory simulation integrates the agent-based modeling paradigm with ideas borrowed from organization theory to construct goal-driven simulations in which participants play specific roles. Similar to a game, but much more organized and structured, participatory simulations are extremely useful for setting up an agent-based model correctly. With proper structure, instruction, and discipline, participants in a participatory ABMS can show much information about agent behaviors. Participatory simulation may be employed for developing and validating agent behaviour models, demonstrating concepts to stakeholders, and the testing of ideas in specially created scenarios.

1.3.3 Agent Based Modeling and Simulation for Epidemic Scenarios

Several different ABMS methodologies have been developed to handle and simulate complex real scenarios about epidemic outbreaks, especially those concerning COVID-19 [23]. According to Dyke Parunak et al. [24], such scenarios can be represented either by agent-based models (ABMS) or equation-based models (EBM). They further argue that:

1. ABMS is particularly well-suited for systems marked by strong localization, distribution, and discreteness in decision-making.

2. EBM is better applied to centrally modeled systems whose dynamics are dominated by physical processes rather than information processing.

Moreover, they conclude that "ABMS might fare even better than EBM if equivalent facilities for building and analyzing SD models are made available.". Rahmandad and Sterman [25] demonstrated stochastic ABMS may outperform differential equation models when there are several unknown parameters and capturing heterogeneity across people and their networks of reciprocatory interactions is important. Ajelli et al. [26] compared ABMS with meta-population approaches in modeling the epidemic in Italy; they concluded that the balance between the two depends upon data availability, supporting the use of hybrid models.

Silva et al. [27] created an individual entity-based simulation to study different COVID-19-related scenarios where the individuals are modeled as mobile particles. Infection happens when two particles come into a predefined contact radius. Social distancing has been modeled by changing the contact radius or adding a momentum term. Hinch et al. [28] developed a model based on household, workplace, and casual contact networks that enabled the researchers and policy-makers to evaluate the impact of lockdown, testing policies, quarantine, and digital/manual contact tracing. The model was validated in the case of an urban area with 1 million inhabitants whose age structure and contact patterns were similar to those of the UK population. In addition, many researchers have supported the use of contact networks (e.g., [29, 30, 31]). Truszkowska et al. [32] presented a high-resolution approach that incorporated individual-level behavior based on real COVID-19 data for New Rochelle, NY, a town of thousands of residents. This model extended the capabilities of classical simulations based on employees, healthcare facilities, schools, and elderly living facilities, among others, and assessed different testing, treatment, and symptomatic infection policies similar to those of COVID-19. Finally, Chumachenko et al. [33] developed an ABMS, which classifies people into four categories, susceptible, exposed, infected, and recovered, based on their status in the current epidemic. These categories enable simulations of epidemic spreading before vaccination is started. People interact with each other and with their environment, where possible dissemination of the illness and transitions between states takes place based on the proba-

bilistic coefficients tuned by the official COVID-19 incidence statistics derived from the Ukraine Health Center. Model simulations demonstrated that self-isolation of patients and contact tracing are the most effective measures to reduce the spread of the epidemic while isolating the whole population is unnecessary isolating 80% of actively infected individuals suffices.

1.3.4 Actor Based Modeling and Simulation

ABMS involves various methods to model simulations that involve independent entities interacting with each other and the environment. Within this broader ABMS framework, Actor Based Modeling and Simulation provide a special kind of interaction handling within a wider ABMS framework by casting a spotlight on the entities or "actors" asynchronous communication interactions.

Actors are autonomous computational units that can interact with other actors by sending and receiving asynchronous messages. Once all responses have been received, they can, in parallel, send more messages, create new actors, and alter their behavior to prepare for the next messages they will receive [34]. Also, actors are suitably endowed with the necessary properties to define and execute the computational agents deployed in multi-agent systems and agent-based modeling simulations [35]. Indeed, actors and computational agents share some properties: i) they are responsive to external events - that is, they exhibit reactivity, ii) they operate independently and exercise control over their internal states - that is, they express autonomy and iii) they communicate by exchanging asynchronous messages with each other, which makes it possible for them to coordinate and cooperate - that is, they exhibit social features [36]. As a result, the existence of different actor-oriented software frameworks may support the development of computational agents in domains where those agents act dynamically, change their behaviors, and should be coordinated or collaborated with through the process of direct interaction. Some other researchers have also proposed some interesting solutions for large-scale ABMS applications. For example, Jang and Agha [37] have proposed an actor-based software infrastructure that ensures the actor's adaptive architecture. This infrastructure supports the construction of large-scale agent-based simulations and exploits some distributed computing

techniques to optimize the distribution of the agents of an application on a network of computational nodes. Apart from this, the software infrastructure employs several optimization methodologies so that the amount of data to be transferred between nodes is reduced, while it will support the distribution and searching of dynamic agents.

Scheutz et al. [38] developed a simulation framework called SWAGES that offers the automated and dynamic distribution of simulations, hence reducing the duration a simulation takes. In particular, SWAGES allows multiple programming languages to be used for the specification of agent models and provides large datasets and analysis methods that can support the automation of results processing. It also features a powerful scheduler that has fault detection and recovery processes built in to make long simulations more robust. In this regard, the authors of Cicirelli et al. [39], based on the Repast software platform [40], proposed a simulation distribution by the actors' implementation. Specifically, they designed a software framework that allowed each application to be divided into separated subsystems, called theaters, gave each subsystem the capability to host a set of actors, and was deployable on every node that composed the application. Wittek and Rubio-Campillo [41] present the ABMS framework called Pandora, which is designed for social scientists using HPC resources and implemented in the C++ programming language.

In addition, Pandora provides an ABMS Python interface to the framework that should make possible the development of ABMS to people with minimal programming background. Pandora splits the simulated environment among multiple computer nodes, each possessing a part of the environment and the agents within that section. Furthermore, data and agents at the boundary between neighboring nodes are copied and sent to the neighboring nodes at every time step of the execution of a simulation to keep track of consistency during the process. Pandora served as a platform for investigating the trade-offs involved in substituting a conventional cluster with a cloud-based solution. Given that the simulation framework necessitates a high-speed interconnection, it is anticipated that this approach would yield suboptimal performance; nonetheless, a cloud cluster is expected to mitigate losses and demonstrate that even an economical cloud cluster can offer significant computational capabilities for the simulations. However, the experimental results indicated that employing

a cloud environment could yield a financially viable alternative. Collier and North [42] developed an ABMS system called Repast HPC. This system is an extension of the Repast framework, developed by North and Collier in 2006. It is designed in C++ and utilizes the Message Passing Interface MPI [43] for high-performance distributed computing on an enormous scale. More specifically, Repast HPC offers a platform where many processes run simultaneously, enabling the distribution of a considerable number of agents over those processes. This is possible because each process has its own scheduler responsible for the processing of local events, while all schedulers are kept synchronized. In the end, Repast HPC allows the creation a shared context from each process that can contain both local agents and remote agents replicated from another process. This solution is bound to enhance simulation efficiency based on the fact that interactions between the local and remote agents operating within the same shared context do not require remote communication. Fan et al. [44] introduced an automated HPC system, called DRAS (Deep Reinforcement Agent for Scheduling), based on deep reinforcement learning methods. More specifically, DRAS uses a hierarchical neural network that offers several key features related to HPC scheduling, including resource reservation and backfilling. Each execution of the DRAS is guided by a different scheduling objective, and during its execution, the system will automatically learn an improved policy via interactions with the scheduling context while it updates its policy due to changes in the workload. The results of its experiments seem to outperform state-of-the-art heuristic and optimization techniques up to 45%. Conclusion: Santana et al., [45] proposed a multivalent simulator that allows complex and large-scale smart city scenarios to be simulated, denoted as InterSCSimulator. Moreover, the proposed simulation paradigm is based on the actor's principle. Every actor conveys a car or a bus that moves within the urban area from a starting point to any other vertex of the city graph. Experimental results showed that the simulator is scalable and easy to use. Moreover, this simulator allows the testing of simulation results by building visualizations and dynamic simulations.

1.3.5 ActoDeS

To efficiently implement an ABMS system using the actor paradigm, the Java framework ActoDeS [46] provides essential tools and structure. ActoDeS is a software environment based on actor architecture. ActoDeS allows the construction of large systems capable of supporting the sharing of various threads among a multitude of actors. An ActoDeS system relies on some set of actors that interact with each other by performing tasks concurrently. The general structure of an actor follows the one already described in the previous paragraph.

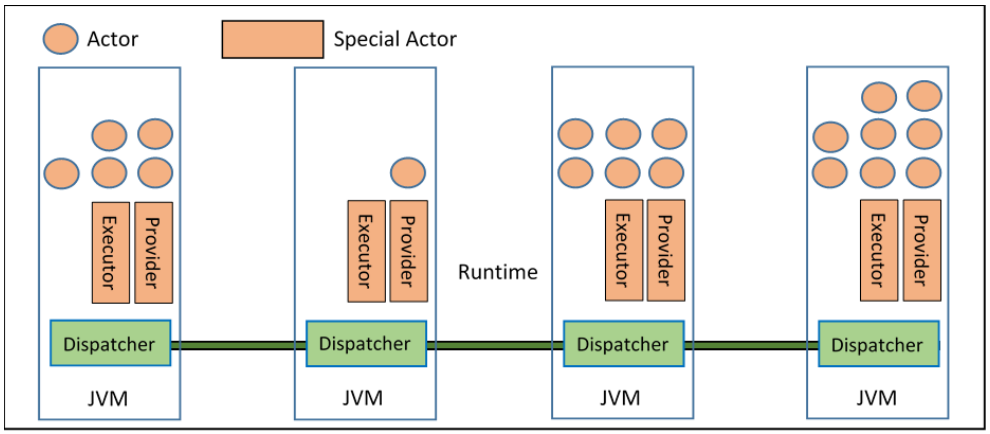

Figure 1.2: Architecture of a distributed application in ActoDeS

According to the application complexity and the provided communication resources, one or more actor spaces can be used to handle the different actors present in the system. An *actorspace* plays the role of a "container" for a set of actors and offers the required services for their execution. An *actorspace* encompasses an *executor* and a *service provider*. The executor manages the concurrency of the actors' execution and the *actorspace*. The *service provider* enables the actors in an application to perform new actions. Figure 1.2 illustrates graphically the architecture of a distributed application in ActoDeS.

ActoDeS has been implemented using the Java programming language. Java's

existing software libraries are used to support concurrency and distribution. In particular, ActoDeS allows the configuration of an application with different types of actors and various runtime components to improve performance.

Actor An actor can be viewed as a light thread running an event loop. The event loop will keep on processing incoming messages. After processing each message, using the return value of its message handler the actor decides whether it wants to continue with its existing behavior, switch to a new behavior, or stop its execution.

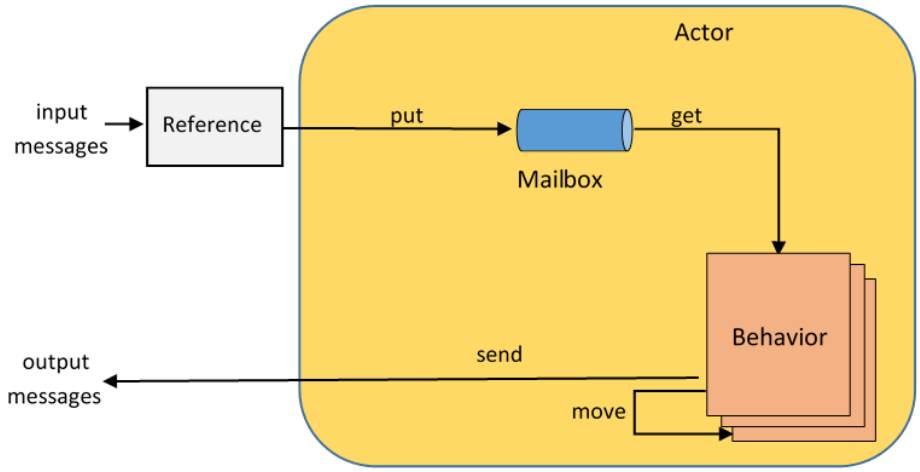

Figure 1.3: Architecture of an actor in ActoDeS

In *ActoDeS* it is possible to implement an actor as either passive or active. While an active actor has his own *thread*, a passive one shares the same *thread* with all other actors in the same execution space.

Reference A *reference* defines a component that acts as a *proxy* for a specific actor with the task of supporting message sending. However, an actor must know the reference of another actor in order to correctly send a message. Specifically, a *reference* can be obtained in two main ways:

Class Name	Description
ActiveActor	Implements active actors. All actions are performed, and then the actor terminates its execution.
CycleActor	Implements passive actors. Provides a method that processes messages received from other actors in the last scheduling cycle.
OldActor	Implements passive actors. Provides a method that processes messages received from other actors before executing the aforementioned method.
OneActor	Implements passive actors. Provides a method that processes the oldest messages present in its queue.
SavableActor	Implements passive actors. Provides a method that processes messages received from other actors in the last scheduling cycle. Additionally, provides a method that gives information about the actor's inactivity cycles.
SharedActor	Implements passive actors. Uses a mailbox that takes messages from the queue shared with all other actors in the actor space. Additionally, provides a method that processes messages received in the last scheduling cycle.

Table 1.2: Types of Actors

- Creating a new actor (in fact, the return of the creation method is a reference).
- Receiving a message from another actor (each message contains the sender's reference or might contain references to other actors).

Additionally, an actor knows four references a priori, which are identifiers of its actor space: the *service provider*, the *executor*, the local *broadcast* service, and the global *broadcast* service. Usually, actors do not send or receive messages from the *executor*, but its reference is useful because it allows interoperability between multiple actor spaces.

A reference has an attribute, called the *actor address*, which allows it to distinguish itself from other references. An address consists of three main components:

- A unique identifier that distinguishes an actor from others within its own actor space;
- A unique identifier indicating its membership in a specific actor space;

- The IP address of the computing node.

Messages A message defines a set of fields intended to contain all the necessary information. Furthermore, each message is unique. Messages coming from the same sender have an identifying code, unique, while messages coming from different senders have a different reference. The table below summarizes all the various fields that a message can contain.

Field Name	**Description**
Identifier	Message identifier
Sender	Reference of the sender
Receiver	Reference of the recipient
Content	Content of the message
Time	Time of sending
Type	Type of message
inReplyTo	Identifier of the message being replied to

Table 1.3: Message Fields

Actor Space An *actor space* is responsible for supporting the execution of actions by its actors. To this end, an actor space relies on three essential runtime components: the controller, dispatcher, and registry, along with two special actors: the executor and the service provider.

The *controller* configures the actor space and controls its activities until the execution completes. It initializes other runtime components and special actors within the actor space.

The dispatch handles the interaction with other actor spaces within this application. This includes establishing connections for sending and receiving of messages, caching remote addresses to outgoing connections, providing an incoming message handler, and sending messages via outgoing connections.

The *connector* works in concert with the dispatcher to keep actor spaces talking to each other. In addition, the connectors act as brokers for all newly created actor spaces; organizing and distributing information. Several predefined connector implementations are briefly described in Table 1.4.

Connector Name	**Description**
ActiveMqConnector	Supports communication between actor spaces using ActiveMQ JMS
JeroMqConnector	Supports communication between actor spaces using ZeroMQ
MinaConnector	Supports communication between actor spaces using the MINA socket library
RmiConnector	Supports communication between actor spaces using Java RMI

Table 1.4: Connector Implementations

To develop a distributed application, actors within an actor space need information about other actor spaces. A centralized object, the *INFO* object, provides relevant data about the current actor space and the components or actors it interacts with. The following table summarizes some of the key methods available for accessing this information.

Method Name	**Description**
getConfiguration()	Retrieves configuration details for the actor space
getBroadcast(Reference c)	Retrieves the broadcast reference for an actor space using an actor's reference
getBroadcasts()	Retrieves broadcast references for connected actor spaces
getBroker()	Retrieves the reference for a service provider acting as a communication broker
getPopulation()	Retrieves the number of actors currently running in the actor space
getProvider(Reference c)	Retrieves the reference for a service provider in an actor space, using an actor's reference
getProviders()	Retrieves references for service providers in other actor spaces
getExecutor(Reference c)	Retrieves the executor reference for an actor space, using an actor's reference
getExecutors()	Retrieves the references for executors in connected actor spaces

Table 1.5: Space Information Methods

The registry is the other core runtime element that allows creating actors, Message Router management, and naming service. It builds references for new actors helps in the delivery of messages from remote actors and allows the dispatcher to determine the final destination reference. An actor can send a message to another only if he knows his reference. A reference from a local actor allows the message to be

delivered directly, while a reference from a remote actor will delegate the delivery to the local and remote actor spaces' dispatchers.

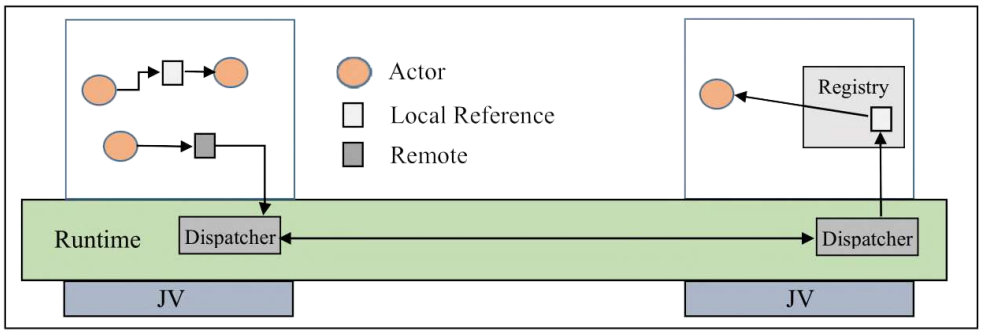

Figure 1.4: Message Delivery Process

Service Provider A *service provider* is a runtime actor that provides the possibility to execute new kinds of actions and offers a set of services to the application's actors. The actors of an application request the execution of such services by sending a message to the *service provider*. Furthermore, every *actorspace* of an application can offer another set of services. Some predefined services available in actor spaces are enumerated in Table 1.6.

Executor An executor is an actor responsible for defining how to execute and manage the operation of other actors in the actorspace. The executor is responsible for creating the initial set of actors and setting up an appropriate actor space. The responsibilities of an executor vary according to the implementation chosen and its actors, especially concerning thread handling. Actors can have a thread of their own either exclusively ("active actors") or shared among all actors in the *actorspace* ("passive actors").

The schedulers for active actors hand much of the work off to the Java runtime environment, while for passive actors it performs the complete processing of their

Service Name	Description	Mandatory
Broadcaster	Transmits messages	Yes
Creator	Creates new actors	No
FileStorer	Provides persistent storage based on files	Yes
Grouper	Manages group communication	No
Logger	Records information about the execution of actors and runtime components	Yes
Mobiler	Creates mobile actors that can move from one actor space to another	No
Namer	Binds an actor to a name, which can be used instead of the reference	No

Table 1.6: Services in an actor space

run. That is the initialization of the actors and iteration calling methods that process messages from their *mailbox*.

The constructor of an *executor* takes as an argument either an instance of the actor's *behavior* in case the initial set consists of a single actor responsible for the creations of the actors and the *actor space*, or an instance of a class that extends the class *Builder* which creates the initial set of actors.

Scheduler Name	Description
CycleScheduler	Operates in workspaces containing only instances of actors of the CycleActor type
OldScheduler	Operates in workspaces containing only instances of actors of the OldActor type
OneScheduler	Operates in workspaces containing only instances of actors of a specific type
PersistentScheduler	Works with actors of the SavableActor type. Additionally, it asks a PersistentRegistry to store actors that are idle and wake them when they receive a new message
PoolCoordinator	Handles execution for actors of the ActiveActor type by utilizing a Java thread pool
SharedScheduler	Operates in workspaces containing only actors of the SharedActor type
TemporaryScheduler	Handles actors of the SavableActor type and requests that a TemporaryRegistry keeps idle actors in memory, waking them when they receive a message
ThreadCoordinator	Operates with actors of the ActiveActor type, creating a Java thread for each created actor

Table 1.7: List of Schedulers

1.4 Cyber-Physical Approaches with AI in Medicine

CPS [47] is a paradigm in which physical processes are tightly integrated with computational elements, often interconnected through networks. In this context, CPS integrates hardware components like sensors, actuators, and medical equipment with software that can monitor, analyze, and react to patient data in real-time. These systems enhance the physical by embedding digital intelligence into them, enabling the automation of several tasks that used to be highly manual and very labor-intensive or impossible for conventional medical equipment to reach.

1.4.1 Medical Cyber-Physical Systems

The advent of Medical Cyber-Physical Systems (MPCS) marks a paradigm shift in how patient care is delivered, with CPS becoming the backbone for innovations such as real-time monitoring, diagnostic automation, and predictive analytics [48].

MCPS are a specialized subset of CPS and play a crucial role in addressing some of the most pressing challenges in modern healthcare. MCPS allows real-time interactions between patients' physical conditions and the cyber world of data analytics, providing more precise medical interventions on time. For instance, medical devices such as ventilators and infusion pumps are recently being integrated into MCPS frameworks to further allow health professionals to remotely monitor and control such devices with greater precision [49]. The primary advantage of such integration is the capability for volume data collection, processing, and response in real time to ensure safety and optimal clinical outcomes for patients.

Moreover, MCPS enables predictive and preventive healthcare due to continuous monitoring. Such systems might find patterns in patients' data that could precede some health problems and thus enable early treatment. For instance, wearable medical devices that are part of the MCPS can monitor vital signs and warn a patient and doctor about a possible problem well in advance of complications arising [50]. This predictive possibility is particularly very valuable in the context of chronic disease management, in which the timely detection of complications may significantly improve the patient's outcomes.

In surgical environments, MCPS has been instrumental in optimizing operating room (OR) processes. Real-time data collection from surgical instruments, patient monitoring devices, and hospital systems ensures that medical teams are better equipped to manage the complexities of surgery. MCPS reduces human errors resulting from human factors by providing continuous feedback and enabling surgical teams to make informed decisions in any critical moment [51]. MCPS has also contributed to postoperative care, where patients can monitor their recovery through devices that connect to physiological changes.

One of the most important aspects of MCPS is its ability to support personalized medicine. By leveraging data analytics and ML algorithms, MCPS can analyze patient-specific data to create personalized treatment plans [52]. These systems enable clinicians to individualize various physiological particularities, including genetic information, life habits, and previous medical history, to make treatments maximally adapted to individual needs. This reduces the risk of adverse treatment reactions, improving overall healthcare efficiency.

1.4.2 Internet of Things in Medicine

The Internet of Things (IoT) is a core component of MCPS, enabling the real-time collection and transmission of data from medical devices and sensors. IoT provides a technological backbone to connect medical devices to hospitals, clinics, and home-based patients for continuous health monitoring. It is an essence in the critical care environment while various data are drawn from multiple sources and need analysis for immediate decisions [53].

One of the most notable applications of IoT in healthcare is wearable devices, such as smartwatches, fitness trackers, and specialized medical sensors. These devices automatically collect physiological data about heart rate, blood pressure, glucose levels, and other vital signs. Integrated use of such wearables in MCPS will enable health care providers to deliver remote patient monitoring and timely interventions through real-time data access [51]. This provides for efficient management of chronic diseases, such as diabetes and hypertension, where continuous monitoring is quite important in preventing complications.

Furthermore, IoT plays a crucial role in perioperative care, especially in the management of operating rooms (OR). IoT devices track the patients in the perioperative setting so that medical teams are assured that a patient's location, status, and requirements are known through the surgical process. Real-time data improves efficiency in operations, reduces waiting times, and enhances the effectiveness of conducting surgeries. IoT-based systems support tracking surgical instrument usage, monitoring sterility levels, and performing inventory management for better utilization of resources in ORs.

IoT also contributes to patient safety through real-time monitoring systems that alert healthcare providers to potential issues. For instance, in hospital settings, IoT-enabled devices can detect changes in patient vitals and trigger alarms for immediate intervention. In intensive care units (ICUs), where patients are critically ill, this continuous monitoring ensures that any deterioration in patient condition is detected early, allowing for rapid medical responses [53]. IoT enhances communication between devices, patients, and healthcare professionals, facilitating a seamless flow of information that improves the accuracy and speed of medical interventions.

The integration of IoT in MCPS extends beyond the hospitals to the level of patients' homes, making health care more proactive than ever. Remote patient monitoring systems enable doctors to monitor the patient's health conditions without necessarily visiting a healthcare facility, thereby reducing hospital congestion and offering individualized care. This is very important for aged patients or those who face problems in mobility, as such patients get quality healthcare from the comfort of their homes [50]. This shift towards home-based healthcare has the potential to transform how chronic conditions are managed and reduce healthcare costs on a large scale.

1.4.3 Digital Twin Technology in Healthcare

Digital Twin technology is emerging as a transformative innovation within MCPS. A DT is a virtual model replicating physical entities, such as human organs or entire medical systems, allowing real-time monitoring, simulation, and analysis. Integrating it with IoT allows real-time data from connected sensors to be mirrored in the digital model. This combination enhances monitoring, analysis, and control, enabling pre-

dictive maintenance and optimized performance. By continuously syncing the physical and digital counterparts, it can be possible to simulate scenarios, detect issues early, and make data-driven decisions. tDT can enable the healthcare professional to do predictive diagnostics and treatment plan simulations before performing these on real patients- a factor that dramatically reduces risks during complex medical procedures. [54].

One of the best applications of DT technology is surgical planning and simulations within healthcare. Building a virtual, real-time look-alike of a patient's body enables surgeons to simulate situations in advance, anticipating how the particular patient may react to various interventions. That is particularly important in complicated surgeries, such as cardiovascular or neurological operations, where minute changes may drastically affect the treatment outcome [55]. Simulated surgeries allow surgeons to discover possible complications, optimize techniques, and enhance patient safety.

For instance, in the field of cardiology, DT models of the human heart have been developed to simulate the behavior of the heart under different conditions. These models allow cardiologists to understand how a patient's heart might react to a specific treatment. Such predictive insights help tailor treatments to the patient's unique physiology, improving both the efficacy of interventions and overall patient outcomes [56].

DTs also play a crucial role in rehabilitation and recovery. By continuously monitoring patients and feeding real-time data into the digital model, healthcare providers can track recovery progress and adjust rehabilitation programs dynamically. For instance, after joint replacement surgery, DTs can simulate the biomechanics of the new joint, helping physiotherapists design personalized recovery plans based on the patient's specific condition [57]. This leads to faster recovery times and improved functionality for the patient.

Beyond this, however, DTs are used to monitor the efficiency of facilities for individual patient treatment. For example, they simulate and manage the flow of patients, medical personnel, and hospital equipment to reduce congestion and enhance the smoothness of all processes. Hospitals can model their operational activities in

virtual versions, including allocating resources such as operating rooms and medical staff and test multiple operational strategies virtually to determine the most efficient ones. This level of operational optimization guarantees that healthcare institutions can deliver quality care at better costs with efficient use of resources.

A particular application of this technology is in personalized medicine. By creating a digital replica of a patient, healthcare providers can better understand how specific treatments will impact that individual rather than relying on general models that assess treatment effectiveness. For example, in oncology, using DTs enables oncologists to simulate how different cancer treatments might affect a patient's tumor, taking into account its genetic characteristics and overall health. [58]. This accuracy helps choose the most appropriate treatment with minimal side effects.

The future of DT technology in healthcare also interlinks with the fast-growing field of AI and ML. Combining AI with DTs allows it to develop self-learning systems that can improve over time by learning from new data. This shall enable continuous refinement of medical procedures, personalized treatment plans, and hospital operations toward increasingly precise predictions and better outcomes.

1.4.4 AI-Driven Cyber-Physical Systems (AI-CPS) in Medicine

CPSs, driven by AI, have become increasingly relevant in the medical domain. These systems integrate computational and physical processes, often in real-time, enabling innovations in health monitoring, diagnostics, and therapeutic interventions. In this context, AI-driven CPS enhances capabilities such as robotic surgery, intelligent medical devices, and personalized treatment plans.

CPSs in medicine that are AI-driven combine the ability of AI to process and analyze vast amounts of data with the capabilities of CPSs to interface with physical surroundings, hence making necessary adjustments according to data analytics. These can also operate independently or semi-autonomously, altering their behaviors based on varying ambient conditions or patient-specific data. For example, medical diagnostics employ AI algorithms that look for patterns in medical images or genomic data that would indicate the presence of a disease. Results are then fed back into the CPS, which adjusts treatment strategies in real-time. Similarly, CPS guides

robotic systems in surgery to perform precise, less invasive procedures guided by AI models trained on huge datasets of prior surgeries [59].

The integration of AI in CPS also holds promise in developing smart prosthetics and rehabilitation devices. For instance, AI can analyze the patient's movement and adjust the prosthetic's behavior in real-time to improve the patient's comfort and functionality. Moreover, these systems can be designed to learn and adapt over time, improving their performance as they gather more data [60].

Shortly, AI-driven CPS will have great potential in health care applications by offering continuous health monitoring with wearable IoT devices that may be integrated within CPS. Such devices, fitted with AI algorithms, monitor critical parameters such as vital signs and detect anomalies by providing early warnings of potential health issues. This model would enable proactive healthcare where interventions are based on predictive analytics rather than reactive treatments [61].

Chapter 2

Proposed Methods for Epidemic Modeling and Simulation

ABMS has become an essential tool for understanding complex systems, especially while dealing with epidemic modeling. The essential elements of ABMS are independent agents representing people or organizations interacting within an environment. Since ABMS can capture the heterogeneous behaviors and social dynamics naturally present in a natural population, it further provides insight into the pattern of the spread of the disease and how to intervene to reduce its effects.

A critical question arises: what are effective strategies for using ABMS to prevent medical issues during epidemics? My contributions are focused on developing and enhancing the *ActoDemic* framework, enabling fine-grained epidemic modeling for large-scale scenarios. This framework aims to enable complex simulations that better reflect the complexity of human behavior and social interactions during an epidemic. The effectiveness of *ActoDemic* is underlined by an application presented: the simulation of COVID-19 diffusion in Lombardy, Italy. This case study validates the framework and shows its capacity to model diverse epidemic scenarios and evaluate the effectiveness of public health interventions. Leveraging *ActoDemic* will enable me to contribute to improved research in epidemic simulation and better-informed public health decision-making.

2.1 ActoDemic Framework

ActoDemic is a cutting-edge framework developed for enhancing the abilities of ABMS in epidemic scenarios. With the increasing need for complex tools to analyze dynamics associated with disease spread, *ActoDemic* addresses some critical challenges related to simulating large populations and capturing intricate interactions driving epidemic behavior.

At its core, *ActoDemic* leverages a distributed architecture that enables the concurrent execution of millions of actors, each representing an individual with unique characteristics and behaviors. This fine-grained approach allows for a detailed exploration of how individual actions and social dynamics influence the trajectory of infectious diseases. By incorporating demographic data, social interaction patterns, and public health policies, *ActoDemic* can simulate realistic epidemic scenarios, providing valuable insights for public health decision-making.

The framework's modular design allows flexibility, allowing researchers to tailor different components according to particular modeling needs. From the simulation of vaccination campaigns and social distancing to the demographic variabilities in disease transmission, *ActoDemic* allows for a strong base in researching manifold epidemic dynamics.

By employing high-performance computing technologies, *ActoDemic* reaches scalability in modeling large-scale epidemics, and it is an essential tool for researchers and public health officials. Moreover, with its application, *ActoDemic* is a tool used to reconstruct past outbreaks and part of active solutions to be better prepared against future public health challenges.

The main reason the actor-based model has been chosen among other approaches is its intrinsic ability to handle onboard, localized, asynchronous interaction of system components. While equation-based models simplify the system's heterogeneity, an actor-based model makes each entity run independently and react dynamically to external actions. This autonomy is an analogy to fundamental epidemic dynamics, where individuals' decisions and behaviors can influence disease transmission.

The scalability and flexibility of actor-based model implementations are remark-

able. By supporting distributed computing, they allow parallel computations and large-scale simulations of millions of entities. This is important in epidemic scenarios where maintaining the granularity of individual interactions will lead to less-more accurate predictions predictions. These levels of detail tend to make other traditional methods, such as system dynamics or metapopulation models, which might come out unable to solve it, or at least one that requires so much simplification that all the realism would be lost.

The actor model also allows modular developments, enabling new component additions with every run without interfering with the whole framework. This flexibility is highly relevant when exploring many epidemic scenarios, like shifting intervention strategies or the effect of demographic changes. For these reasons, an optimal trade-off between computational efficiency and the realization of adequate detail in the simulation has been chosen as the actor model.

2.1.1 Properties

ActoDemic is built upon core principles designed to support scalable and flexible simulations, providing a foundation for modeling complex systems through customizable and distributed actor-based architectures. Moreover, The *ActoDemic* framework is built on top of the ActoDeS system, inheriting its core properties to handle large-scale, distributed simulations efficiently. By leveraging the actor-based architecture of ActoDeS, *ActoDemic* benefits from the autonomous, concurrent execution of agents, allowing it to model complex epidemic scenarios with high scalability. The actor-oriented nature of ActoDeS enables ActoDemic to manage interactions between agents while efficiently maintaining low computational overhead. Additionally, *ActoDemic* uses the dynamic communication and flexible agent distribution features of ActoDeS, ensuring robust and adaptive simulation performance, even in computationally demanding environments. This foundation suits *ActoDemic* for detailed epidemic modeling and real-time simulation tasks.

The base unit in this framework is represented by actors, which correspond to the entities within the system being modeled and simulated, depending on the target application. In case the simulation needs to be fine-grained, a large number of con-

current actors is required. ActoDeS provides the infrastructure for such concurrent agents to spread their computational load over a set of nodes. Since the implementation of ActoDeS is in Java, *ActoDemic* operates on multiple operating systems. At the same time, quick development and prototyping are enabled by several features: automatic serialization and a wide range of libraries in Java.

ActoDemic creates at least one actor space in each computation node used in a simulation. One of the significant problems in designing ABMS systems for large-scale systems is the issue of fitting the whole set of agents into one cluster node. When modeling spreading phenomena, behavior must first be defined for entities, their interactions, and the characteristics of the spread itself by the developer. *ActoDemic* offers a default actor to accommodate diverse spreading phenomena that can be extended by adding new behaviors implemented by Java class definitions. Such a default actor for modeling system entities will be referred to as a Base Spreader (BS). For example, if *ActoDemic* is used to simulate the spread of a pathogen, every person is represented and implemented as a BS. A BS has a set of default attributes that can be refined to suit the particular model being built:

- **Identification number**: A unique identifier for each individual.
- **Community or cluster membership**: In spreading simulations, grouping entities into clusters and adjusting the interaction bias within and between clusters is often necessary. Entities can be grouped based on geographic location or social communities. If clustering is unnecessary, all entities can belong to a single partition.
- **Interaction level**: Different interaction rates can be defined among entities to model more complex and detailed scenarios.
- **Current phase**: An indicator showing the entity's stage in the contagion process.
- **Spreading reducer**: A mechanism that mitigates the spread, such as a vaccine or protective equipment.

2.1.2 A Modular Approach

ActoDemic highlights the modularity of the software for all aspects related to the execution and management of the simulation. However, a modular structure will be easier to adapt to whatever epidemic model might be proposed and more accessible to set up.

Figure 2.1 shows the modular architecture. The modular architecture can be divided into four modules:

- Agents initialization and distribution module (**AID**)
- Spreading Management (**SM**)
- Synchronization and Message Passing (**SMP**)
- Utility to generate reports and evaluate the simulation (**RG**)

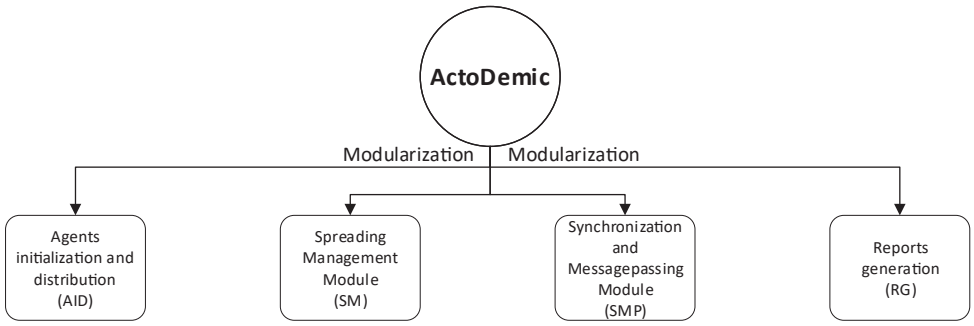

Figure 2.1: Framework modules

AID module

The AID module is responsible for the distribution of the simulation on several nodes and will enable the scalability of large-scale simulations. It deals with the initialization of all the entities. ActoDeS, gives the possibility to create and manage actors and to exchange messages between them. The very first action performed by the AID module is the initialization of the needed actor-spaces. For every actor-space, AID

assigns a separate thread, shared between agents belonging to the same space. This means that every entity will be a passive actor that will share its thread with other actors in the same actor-space.

In addition to initializing actor-spaces, AID creates the BSs (actors) and distributes them across the available nodes, following specific criteria that can be adjusted by the developer to meet the needs of the simulation. These criteria include:

a. Partitioning entities based on their respective cluster or community, if more than the default single cluster is defined;

b. Dividing the population into equal-sized subsets, depending on the number of actor-spaces involved in the simulation.

To manage the base spreaders efficiently, the AID module also initializes the ActoDeS schedulers and managers within each actor-space, as discussed in 1.3.5. The AID module is specifically tailored to each application and is responsible for launching all distributed components of the system. Within each actor-space, the manager creates a subset of agents for its computational tasks and synchronizes the execution of the simulation on that node with others. Additionally, the last manager to be created assumes the role of the "Master," overseeing the overall coordination. It is also possible to define multiple actor-spaces on each computational node for greater flexibility.

The default population distribution algorithm, outlined in Algorithm 1, assigns actors to the N available actor-spaces based on their unique identification numbers. This algorithm, executed by the AID module, ensures an even distribution of actors across the nodes. Each subset of actors generally includes agents within the range $(n-k)\cdot p-1$ to $(n-k+1)\cdot p-1$, where n is the total number of partitions, k represents the current partition, and $p = \frac{population}{N}$ is a constant that defines the size of each subset.

Algorithm 1 Pseudo-code for distributing the whole agents' set across N actor-spaces

```
1: 𝒩 ← GetTotalActorSpaceNumbers()
2: ℛ ← GetAllReferences()
3: function BUILDPOPULATION(𝒩, ℛ)
4:     master ← amITheMaster()
5:     Begin ← ∅
6:     End ← ∅
7:     if master is True then
8:         for (i = 0; i < 𝒩; i++) do
9:             𝒮 ← ((𝒩 − i) · p − 1)
10:            ℰ ← ((𝒩 − i + 1) · p − 1)
11:            SendMessage(ℛ[i], (𝒮, ℰ))
12:        end for
13:    end if
14:    loop
15:        Wait a message from the master Actor-space                ▷
16:    end loop
17:    Begin, End ← ProcessMessageFromMaster()
18:    CreatePopulation(Begin, End)
19: end function
```

where:

- $\mathscr{N}$ the total number of actor-spaces to be created;
- $\mathscr{R}$ is a set that contains all the Actor-Space references (unique system-wide id that is needed to reach an actor and communicate with it);
- *master* is a Boolean value that specifies whether an Actor-space acts as a master or not;
- *SendMessage*() is a function that communicates to an Actor-space which population subset it has to manage.

- $Begin, End$ define the range of the actors to be created and managed ;
- $CreatePopulation()$ divides the population into subsets

In *ActoDemic*, three key entities inherited from ActoDeS play crucial roles: the Broker, the generic node, and the Initiator. The Broker acts as the middle layer that receives messages from the producers and guarantees their delivery to the consumers. On the other hand, the Initiator acts as the master node responsible for managing the creation of agents in each of the actor-spaces. Finally, the generic node is solely responsible for managing the actors within its actor-space, without any other duties.

Additionally, AID includes a sub-module that integrates *ActoDemic* with the Slurm Workload Manager (SLURM) and the MPI protocol [62]. The MPI protocol helps distribute and manage actor-spaces across computational nodes. These features make *ActoDemic* suitable for High Performance Computing (HPC) environments, where SLURM and MPI are commonly used standards.

Further details are provided in algorithm 2.

SM Module

ActoDemic applies principles from Network Science to model the interactions between entities within a system. The SM module is responsible for defining the structure and nature of these interactions. Each actor-space autonomously manages its actors, along with the distinct interactions that occur between them.

Within this framework, each actor is conceptualized as a node within a graph, where outgoing links represent the entities the node interacts with, and incoming links correspond to entities interacting with the node. Graph theory provides a method to analyze the distribution of interactions by studying the degree distribution of nodes, which is a critical factor influencing the dynamics of phenomena such as epidemic spread. *ActoDemic* simplifies the process of configuring these interactions by allowing users to select a predefined interaction distribution, offering options such as power-law, log-normal, exponential, Gaussian, and Poisson distributions.

Moreover, the SM module enables the partitioning of the agent population into clusters, where interaction probabilities are higher within the same cluster and re-

Algorithm 2 Pseudo-code to distribute X actor-space across N computational nodes

```
1: 𝒩 ← GetNodesNumber()
2: 𝒯 ← GetTaskPerNodesNumber()
3: function SPREADACTORSPACES(𝒩, 𝒯)
4:     N_JOB ← 𝒩 · 𝒯
5:     BrokerIp ← ∅
6:     for (i = 0; i < N_JOB; i++) do
7:
8:         if i == 0 then
9:             BrokerIp ← SetBrokerIp()
10:            LaunchActoDemic(broker, BrokerIp)
11:        else if i == (N_JOB − 1) then
12:            LaunchActoDemic(initiator, BrokerIp)
13:        else
14:            LaunchActoDemic(node, BrokerIp)
15:        end if
16:    end for                ▷ The For cycle is managed by MPI on SLURM
17: end function
```

where:

- $\mathscr{N}$: Number of computational nodes;
- $\mathscr{T}$: Number of tasks using a single CPU on every node;
- *N_JOB* Number of actor-spaces that will be created;
- *LaunchActoDemic*() is the function that launches an *ActoDemic* instance and it takes two arguments.

duced between different clusters, reflecting real-world social or geographical groupings.

When two individuals engage in interaction, a procedure is initiated that simulates the process of infection. A default epidemic diffusion model was implemented

using the SEIR mathematical framework. The compartments within the model are fully customizable, allowing for significant flexibility in simulating infection dynamics. Individual compartments can be disabled, thereby altering the progression of the infection. Additionally, intermediate compartments can be introduced to incorporate additional stages into the simulation process, and the duration between various phases can be finely adjusted.

Furthermore, the SM module incorporates a mechanism to regulate the contagion dynamics through a variable known as "Transmission Probability" (TP). This variable ranges from 0 to 1 and quantifies the likelihood of a successful contagion event occurring between two entities.

ActoDemic also provides the capability to define and specify behaviors in the event of a re-infection after an individual has recovered, further enhancing the model's realism and adaptability to different epidemiological scenarios.

SMP Module

The SMP module manages message exchange and time synchronization within the system.

ActoDemic employs a straightforward scheduler known as the "CycleScheduler," which is provided by ActoDeS. This versatile scheduler can be used across various applications, including Agent-Based Modeling and Simulation (ABMS). It oversees the passive actors within each actor-space and cyclically executes the same actions until the simulation concludes:

1. It sends a "step" message to all agents and increments the "step" value, triggering the transition from one epoch to the next.
2. It performs an execution step for all agents.

In this architecture, each actor-space is modeled as an actor, meaning that one actor cannot directly access the internal state of another actor. Consequently, if one or more Base Spreaders from a particular actor-space interact with Base Spreaders from a different actor-space, it is essential to communicate this interaction to both actor-spaces. This is achieved through a message exchange system.

A pivotal property of the actor model is that message exchanges are asynchronous, which can create synchronization challenges. An actor-space may advance to the next epoch without having received all necessary information from the other actor-spaces. Therefore, an actor-space must gather information from all other actor-spaces before proceeding to the next epoch. The SMP module addresses this issue by implementing a "barrier" mechanism that prevents actor-spaces from transitioning to the next epoch until all message exchanges between actor-spaces are complete.

Information routing and management can vary depending on the specific model that the user wishes to implement. However, there is a fundamental structure that can be outlined as follows. Certain information is disseminated across all partition types, while other information is restricted to the master.

The content of a generic message can be summarized as follows:

1. Information sent to all partitions:
   - **BS' Interaction Information**: Notifies the actor-space of interactions between two or more BS.
   - **Status Change Notifications**: specifies which BS must change their status due to triggering events.
   - **Closure Notices**: indicates which actors need to terminate their cycles.
   - **Statistical Data**: provides data necessary for generating reports.
   - **Synchronization Messages**: serves as a "barrier." An actor-space can advance to the next epoch only after receiving all synchronization messages from the other involved actor-spaces.
   - **End Signal**: communicates the conclusion of the simulation process.
2. Information sent exclusively to the master:
   - **Partial Summary Report**: contains data required for generating final reports.
3. Information sent only by the master:

- **Shutdown Commands for Base Spreaders**: the master determines which Base Spreaders should terminate their execution based on specified criteria.

RG Module

The RG module facilitates report generation throughout the simulation process. Users can enable or disable report generation and specify the frequency of report generation.

Typically, after each epoch, every actor-space produces a report that refers to its specific partition and actors, known as "intermediate reports." Additionally, these intermediate reports support a "Save&Load" functionality, allowing the simulation process to be restarted from a specific epoch. To enable this feature, reports must include specific information for each Base Spreader within the actor-space:

- The Identification Number
- Belonging in any type of social cluster
- The interaction level
- The current epidemic phase
- The duration of the epidemic phases
- The status of any active Spreading reducer
- A record of past interactions with other Base Spreaders

At the end of the simulation, the master node generates a final report summarizing the most significant and relevant information. Specifically, it provides a summary for each simulation epoch, detailing the number of individuals in each specific epidemic compartment.

Moreover, the RG module is responsible for the initial configuration. This capability allows *ActoDemic* to support an external configuration file, enabling users to customize and specify all the properties listed previously.

2.2 Fine-Grained Modeling of COVID-19 Spread in Lombardy

After building and refining the framework in all its aspects, a practical use case was required to evaluate its functionality. To validate *ActoDemic*, the focus was placed on the spread of COVID-19. The social interactions, which are the primary drivers of COVID-19 transmission, can be effectively modeled by appropriately adjusting the parameters used by *ActoDemic*.

Additionally, one of the main objectives was to assess *ActoDemic*'s scalability by testing it in large-scale simulations. Simulating millions of agents on a High-Performance Computing (HPC) system is critical to demonstrate the framework's ability to handle computationally intensive scenarios and ensure efficient performance in such environments.

Since late 2019, the rapid spread of COVID-19 has led governments worldwide to impose restrictive measures like lockdowns and social distancing to mitigate transmission. However, many outbreaks proved challenging to predict and manage due to the virus's characteristics, such as asymptomatic carriers, long incubation periods, and limited early knowledge about SARS-CoV-2 transmission dynamics. This highlights the importance of accurate models to simulate various outbreak scenarios.

Each agent in the model represents an individual, characterized by daily social interactions based on factors like age, occupation, and sociability. This fine-grained approach allows the model to simulate both scenarios with and without lockdowns or social distancing measures.

In this context, modeling complex and large-scale scenarios becomes essential for public health and policy-making. COVID-19 transmission is largely driven by social contact, especially in environments like workplaces, schools, public transport, and recreational spaces, where physical distancing is harder to maintain. *ActoDemic* enables detailed simulations that can guide preventive strategies.

The model was implemented on the University of Parma's high-performance computing infrastructure, leveraging the necessary resources for scalable computation. To validate the model's robustness in real-world settings, COVID-19 outbreaks

in Lombardy during the first and second waves were simulated from January to April 2020 and August to December 2020.

2.2.1 Simulation Architecture and Execution Process

In the simulator's architecture, each actor represents an individual with specific attributes such as age, province of residence, and sociability level. The behavior of each agent is influenced by probabilistic parameters, which will be explored afterward. The simulation unfolds over several phases referred to as "epochs," where each epoch represents a day in the simulation. Agents can modify their behaviors depending on the current epoch, for example, shifting between normal periods and lockdown scenarios.

Agents are distributed based on two criteria:

1. Grouping by their province of residence;
2. Dividing into equal-sized subsets, depending on the number of actor spaces employed in the simulation.

The simulation runs across a network of computational nodes coordinated by a set of schedulers and managers, which leverage ActoDeS's passive actor framework to handle large-scale simulations. Each manager is responsible for creating a subset of actors for its node and ensuring synchronization between nodes. One manager takes the role of "master", tasked with partitioning the agents involved in the simulation and distributing this information to the other managers to create the necessary actors under their control.

The simulation process follows these steps:

1. The master manager creates agents and provides the information for other schedulers to create their respective subsets.
2. Each manager generates the actors assigned to its computational node.
3. The simulation then proceeds in repeated cycles:
    (a) Managers exchange synchronization messages and wait for responses

from all other nodes.

(b) Schedulers execute a step for all actors under their control.

(c) After each step, schedulers send an "end step" message to their actors and other managers.

In this setup, each actor space operates as a manager, with the last-created actor space playing the role of the master.

ActoDeS's "CycleScheduler" manages the simulation, particularly in Agent-Based Modeling and Simulation (ABMS) scenarios. This scheduler is responsible for controlling passive actors within each actor space, repeatedly performing the same tasks until the simulation completes:

1. Sending a "step" message to all agents, advancing the epoch and moving the simulation forward;

2. Performing an execution step for all agents within that epoch.

Since the simulation involves interactions between agents across multiple computational nodes, distributing them over these nodes can introduce communication overhead. To address this, the frequency of interactions is reduced, and multiple interactions are merged when possible. To ensure efficiency, agents exchange data and synchronize their state at the end of each epoch, along with the synchronization messages exchanged between the actor spaces.

2.2.2 Social Behavior Modeling

Data from official sources was collected to model the spread of COVID-19 in Lombardy [63, 64], focusing on daily new cases and fatalities reported over time. In addition to this, demographic information, such as population size and age distribution for the region, was incorporated into the analysis [65].

For simulating social interactions, studies conducted by the Italian National Institute of Health were relied upon [66]. These studies considered the average number of contacts a person should have daily. This information is summarized in Table 2.1,

aimed to understand how respiratory infections spread. The average number of contacts by age, the value used in this study, refers to the 'total' column. Meanwhile, Table 2.2.2 shows the population distribution by age in the Lombardy region; these data were provided by the National Institute of Statistics (*ISTAT*) [67].

Age	Total	Home	School	Work	Transport	Free time
0-4	16.54	4.49	5.27	0	0.98	5.81
5-9	20.49	4.61	8.87	0	1.12	5.9
10-14	27.38	4.43	11.98	0.2	1.35	9.42
15-19	29.28	4.59	13.22	0.05	1.74	9.7
20-24	22.15	3.51	1.17	4.49	0.96	12.03
25-29	21	3.47	2.23	5.21	1.13	8.96
30-34	18.03	3.55	0.85	3.92	0.76	8.962
35-39	21.25	4.38	0.68	7.78	1.05	7.37
40-44	22.35	3.88	2.53	7	0.67	8.27
45-49	19.27	2.99	2.61	8.24	0.88	4.57
50-54	22.3	2.75	5.54	8.05	0.52	5.43
55-59	18.27	2.88	1.41	4.6	0.68	8.68
60-64	18.43	3.28	1.07	6.05	0.87	7.16
65-69	12.74	3.1	0.55	0.48	0.95	7.66
70+	10.55	3.24	0.06	1.04	0.22	5.99

Table 2.1: Average number of interactions by age

To build COVID-19 compartments, early research on the incubation period was used [68], COVID-19 outbreak characteristics [69], and age-related susceptibility to the virus [66]. Modeling the lockdown in Italy required data about "*essential workers*" [70] and the proportion of the population using protective equipment [71], as well as the effectiveness of these devices [72].

During the initial week of the simulation, a set of regular contacts was established for each individual. This group represents the people they typically encounter daily: family members, friends, and colleagues. Essentially, these are the familiar faces that fill their day-to-day interactions.

Age	Male	Female	Total	
				%
0-4	205.299	194.557	399.856	4,0%
5-9	239.046	224.761	463.807	4,6%
10-14	254.391	238.914	493.305	4,9%
15-19	247.380	230.340	477.720	4,7%
20-24	253.912	229.596	483.508	4,8%
25-29	262.010	250.130	512.140	5,1%
30-34	282.410	275.681	558.091	5,5%
35-39	313.625	306.736	620.361	6,1%
40-44	369.540	359.909	729.449	7,2%
45-49	420.573	409.422	829.995	8,2%
50-54	425.683	420.177	845.860	8,4%

Age	Male	Female	Total	
				%
55-59	374.754	381.832	756.586	7,5%
60-64	305.783	324.981	630.764	6,2%
65-69	269.084	295.617	564.701	5,6%
70-74	256.056	292.193	548.249	5,4%
75-79	200.480	250.519	450.999	4,5%
80-84	157.532	222.870	380.402	3,8%
85-89	80.444	149.594	230.038	2,3%
90-94	26.783	73.420	100.203	1,0%
95-99	4.737	20.880	25.617	0,3%
100+	248	2.070	2.318	0,0%
Totale	4.949.770	5.154.199	10.103.969	100,0%

Table 2.2: Population age distribution in Lombardy

It was found that each person engages with a mix of both familiar and new contacts. According to Table 2.1, about 65% of an individual's daily interactions happen with those regular contacts, while the other 35% involve people they don't usually see. The regular contacts of focus are based on the Home, School, and Work categories, tailored for each age group.

2.2.3 Estimating Sociability Levels in Social Networks

Individual interactions can be theoretically mapped and analyzed using network science techniques. In this model, each individual is represented as a node within a social graph, where the outgoing and incoming edges indicate the individual's contacts and those of other individuals with whom they interact.

Analyzing the distribution of social interactions is crucial for understanding how epidemics spread. After evaluating various hypotheses, The analysis was ultimately focused on a *power-law* distribution. This model has proven effective in representing many real-world scenarios, particularly in the context of social interactions.

Modern society features specific locations where individuals gather, such as offices, schools, and parks. These locations form significant points of contact known as *hubs* or *super-spreaders*. A hub is defined as a node with a higher degree of interac-

tion than other nodes. The degree of a node reflects the number of edges connected to it. The degree distribution is described using the function $p_k : \mathbb{N} \to [0,1]$, which associates a degree k with the probability of a node having that degree:

$$p_k = \frac{N_k}{N} \tag{2.1}$$

where N represents the total number of nodes, and N_k denotes the number of nodes that have k as their degree.

In a typical *power-law* distribution, nodes with relatively high degrees are more likely to exist. This characteristic makes the power-law distribution suitable for modeling our social network. The degree of distribution of the graph created by social interactions is the focus of the analysis to estimate the sociability rate.

The main goal is to achieve a *power-law scale-free* distribution [73], which is distinguished by the presence of numerous hubs that generate a "long tail" in its graphical representation. A distribution is classified as *scale-free* if it maintains the same shape when scaled in size.

To analyze the distribution, the latest methodologies in curve fitting were referenced [74], employing the Likelihood Ratio test and the Kolmogorov-Smirnov distance to identify the probability distribution that best aligns with our data, specifically searching for a power-law distribution. The choice was made to proceed with the *Complementary Cumulative Density Function* (CCDF) for this analysis, as CCDFs are often preferred for visualizing heavy-tailed distributions. A CCDF measures the probability that a node's degree exceeds a reference value x:

$$p_k = \sum_{q=k+1}^{\infty} p_q \tag{2.2}$$

If p_k follows a *power-law* trend, then the cumulative distribution scales according to the law:

$$p_k \sim K^{-\gamma+1} \tag{2.3}$$

Individuals in the model are characterized by their degree of social interaction, which may vary significantly. Each person is identified by their *Sociability Rate* (SR),

which can take on four distinct values: *high, medium, low,* and *quarantine*. This classification determines how many people they can meet in a single day of simulation. Interactions are generated randomly, forming an interaction graph.

Individuals with a *high* degree of sociability interact with more people than average, while those with a *medium* degree meet exactly the average number. Conversely, individuals with a *low* degree have fewer interactions than average. In our population, 20% of agents exhibit *high* or *medium* degrees of sociability, while the remaining 80% possess a *low* degree.

Three social interaction multipliers were employed to adjust the number of contacts made each simulation day to achieve this distribution. The multiplier for individuals with a *medium* degree is 1. A grid search was conducted over the parameters to identify the optimal multipliers for *low* and *high* sociability rates. The objective was to find the combination that produces a power-law distribution with scale-free characteristics, Table 2.3. The multiplier range for SR = *low* is $[0.1, 0.9]$, while for SR = *high*, it is $[1.1, 1.9]$. The unique case of individuals in *quarantine* will be discussed later.

The CCDFs obtained from each configuration tested during the grid search were analyzed. Figure 2.2 presents a selection of these results and a comparison to a power-law distribution.

To select the optimal configuration, three different parameters were considered:

– *LR (Likelihood-Ratio) test*: This test compares two candidate distributions. A positive result indicates that the data are more similar to the first distribution, while a negative result suggests a closer alignment with the second distribution. The significance level of the result (p-value) refers to the null hypothesis that the two distributions are identical.

– *KS (Kolmogorov-Smirnov) test*: This method measures the distance between a candidate distribution and the empirical data, allowing us to compare a sample against our reference distribution, which is the power-law.

Table 2.3 summarizes the results obtained for the selected multipliers.

Based on the findings summarized in Table 2.3, the configuration *0.2 - 1 - 1.8* was

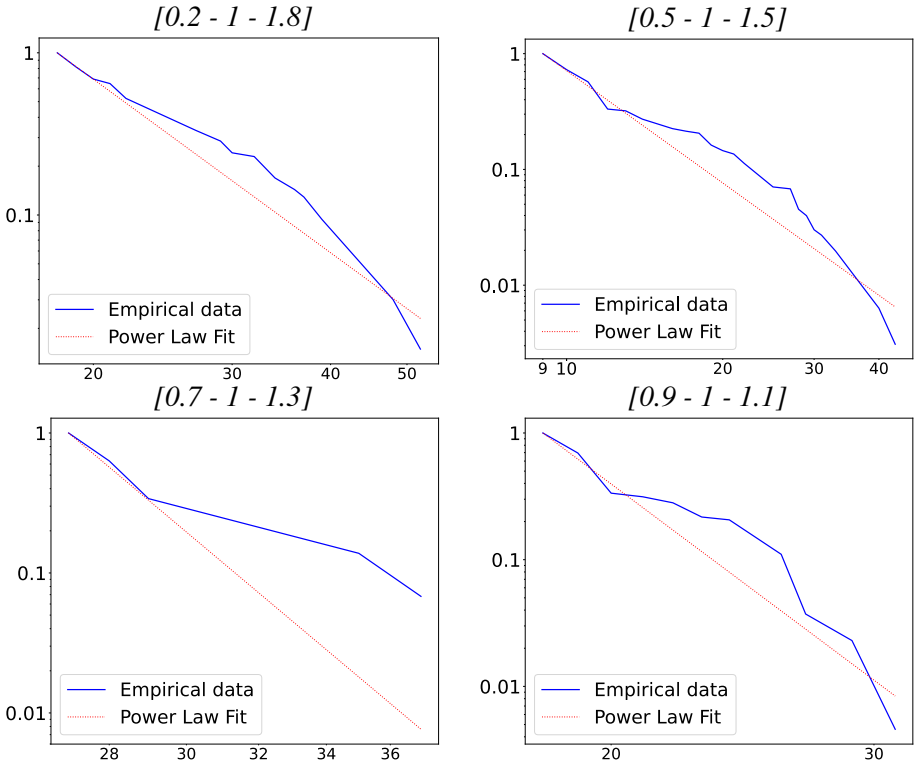

Figure 2.2: Complementary cumulative density function according to interaction multipliers: power-law fit in red, empirical data in blue.

selected. This combination of multipliers is optimal as it yields the highest likelihood ratio and results in a power-law exponent α of 2.64, with x_{min} set at 4. A distribution is classified as *scale-free* when $\alpha < 3$.

2.2.4 Epidemic Modeling

The model used for simulating the spread of COVID-19 is an adaptation of the widely recognized SEIR (Susceptible-Exposed-Infectious-Recovered) model [75]. Therefore, the SEIR compartments were modified by introducing an additional stage: *Positive*, which lies between Infectious and Recovered. This new compartment specifically reflects COVID-19 cases, capturing individuals who have been officially diag-

Multipliers			LR test	LR p-value	KS test
0.1	1	1.9	1.99	$\simeq 0$	0.11
0.2	1	1.8	364	$\simeq 0$	0.21
0.3	1	1.7	-1.46	0.145	0.116
0.4	1	1.6	-150	$\simeq 0$	0.122
0.5	1	1.5	-403	$\simeq 0$	0.093
0.6	1	1.4	-465	$\simeq 0$	0.113
0.7	1	1.3	-205	$\simeq 0$	0.12
0.8	1	1.2	-301	$\simeq 0$	0.09
0.9	1	1.1	-803	$\simeq 0$	0.127

Table 2.3: Grid-search results of the sociability rates considering the most promising distribution candidates. When computing the Likelihood, the first candidate is always the Power-law and the second the Log-normal distribution.

nosed with the virus via throat-swab testing.

This distinction affects behavior modeling and assumes that individuals classified as *Positive* will isolate themselves to limit further transmission. In contrast, those in the *Infectious* state might be unaware of their infection, continuing regular social interactions and contributing to the virus's spread. This difference between *Infectious* and *Positive* is particularly significant in modeling the early phases of the pandemic when testing resources were limited, leading to underestimation of the true number of infections.

Given that the approach is agent-based, differential equations were avoided to dynamically define individual behaviors. In the simulation, individuals transition through several stages after contracting the virus. Initially, everyone is in a *Susceptible* state, capable of being infected by an infectious individual. Upon infection, they move to an *Incubation* stage, remaining there for a specific period before becoming infectious. During this infectious period, they can transmit the virus to others. When this phase concludes, individuals enter the *Positive* stage, after which they either recover or die, following Lombardy's actual death rate trends. Once recovered, individuals

gain immunity to the virus.

The duration of each phase is modeled based on real-world data: the incubation period lasts between 7 and 14 days, the infectious period between 3 and 7 days, and the positive state between 14 and 30 days [68, 69]. A visual representation of the infection cycle is shown in Figure 2.3.

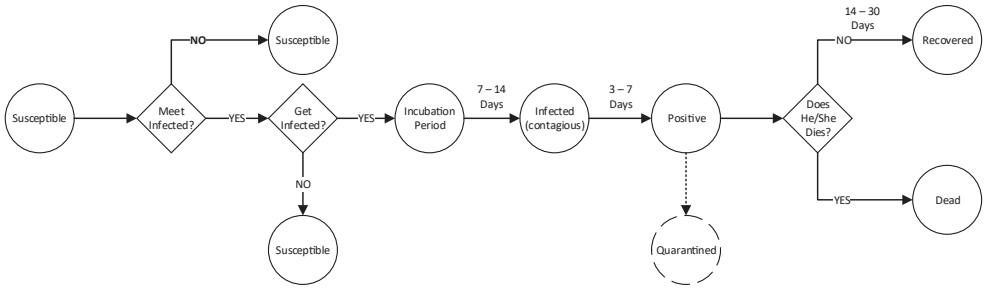

Figure 2.3: Stages of the infection cycle for COVID-19

The susceptibility to infection varies based on age [66]: individuals aged 0-14 have a 31% susceptibility rate, those aged 15-64 have a 47% susceptibility rate, and those aged 65 and above are fully susceptible to the virus.

2.2.5 Exploiting HPC

To simulate the population of Lombardy, which consists of 10 million virtual agents, a High-Performance Computing (HPC) system was exploited. Each agent, referred to as a BS, represents an individual, and the simulation framework distributed these agents across several computational nodes.

The HPC infrastructure used for the simulation is provided by the University of Parma. Each simulation involves dividing the 10 million agents into smaller groups, each managed by a different computational node. The nodes are organized as shown in Figure 2.4. The system involves 32 computational nodes, each equipped with two INTEL XEON E5-2683v4 2.1GHz processors (totaling 32 cores) and 1024 GB of RAM. For the Lombardy region, these 32 nodes handle approximately 137,500

agents each.

Each computational node is assigned a subset of agents (Base Spreaders) to simulate their actions and interactions. The system's architecture ensures that all computational nodes operate in parallel but remain synchronized through a master node. The master node manages the overall coordination and ensures data consistency across nodes by communicating with the managers on each node.

At the end of each epoch, every computational node reports the current state of its agents to the master node. This includes updating information such as the number of active agents, their state changes, and any other necessary synchronization data. The master node ensures that all nodes stay synchronized for the next epoch, distributing any global information required for the next stage of the simulation.

The simulation for Lombardy, with its 10 million agents, typically takes 6 hours and 30 minutes to complete, using around 500 GB of RAM across the entire HPC system.

During each epoch, computational nodes exchange information to ensure consistent state updates across the simulation. The data exchanged between nodes include:

- Information about agents who have changed states (e.g., from healthy to infected).
- The number of agents expected to move to different health stages (e.g., incubation).
- The total number of active agents remaining in each node.
- A summary report is sent from each node's manager to the master node at the end of each epoch.

This communication and synchronization process ensures that the entire simulation remains cohesive and that each node is aware of the global state of the population. The master node collects the necessary data and redistributes relevant information for the next epoch. Using 32 nodes and the ActoDeS framework allows the simulation to scale efficiently while maintaining accuracy and performance.

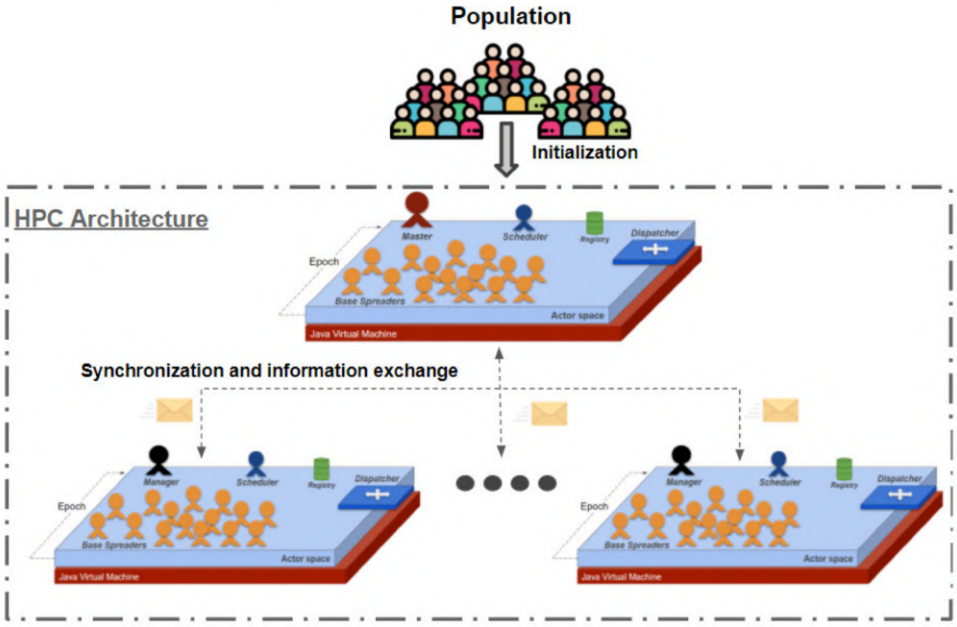

Figure 2.4: The simulator's architecture on the High-Performance Computing (HPC) system, dividing the population into several actor spaces distributed across different computational nodes.

Algorithm 3 shows the pseudocode illustrating the message exchange and overhead reduction mechanism.

For the Lombardy region, this high-performance setup ensures that large-scale simulations, involving millions of agents, can be conducted efficiently within a reasonable time frame.

2.2.6 Modeling Virus Transmission and Contagion Dynamics

Transmission probability, which represents the likelihood of spreading the virus, depends on several factors, such as proximity to infected individuals and the environment (e.g., ventilation). For simplicity, a parameter named *Covid-19 Transmission Probability (CTV)* was introduced to encapsulate these complexities, which aligns

Algorithm 3 Pseudo-code to manage agents across different computational nodes and synchronized through the master node

```
1: Master ← MasterManager()                          ▷ Creates the master managers
2: for AS ∈ ActorSpaces do
3:     Managers ← Master.CreateManager()    ▷ Create a manager for each actor space
4: end for
5: for M ∈ Managers do
6:     M, CreateBaseSpreaders()   ▷ Each Manager instantiates its portion of BSs
7: end for
8: repeat
9:     for each Manager do
10:        for each BaseSpreaderinManager do
11:            Manager.gatherBSsMessages()      ▷ Managers aggregate data from BSs
12:        end for
13:        Manager.sendMessage() ▷ Each Manager sends its message to the other managers
14:        Manager.sendSynchMessage()   ▷ Each Manager sends a synch message to the other managers and waits for the response
15:        Manager.waitResponses()
16:    end for
17:    for each Scheduler do
18:        Scheduler.executeStep()
19:        Scheduler.sendEndStepToAll() ▷ Each scheduler performs an execution step of all its actors and then sends an "end step" message to all
20:    end for
21:    this
22: until EndSimulation
```

with existing approaches. However, as this parameter is not well-documented in the literature, it was estimated empirically using data-driven methods (detailed in Section 2.2.8).

In this stochastic model, only actors in the infectious or positive stages can spread the virus. Each actor has a transmission probability (*TP*), representing their ability to pass on the virus. The probability of contagion (*CP*) when a susceptible individual encounters an infectious one is a function of both their transmission probability and the mask protection levels, as shown in Figure 2.5.

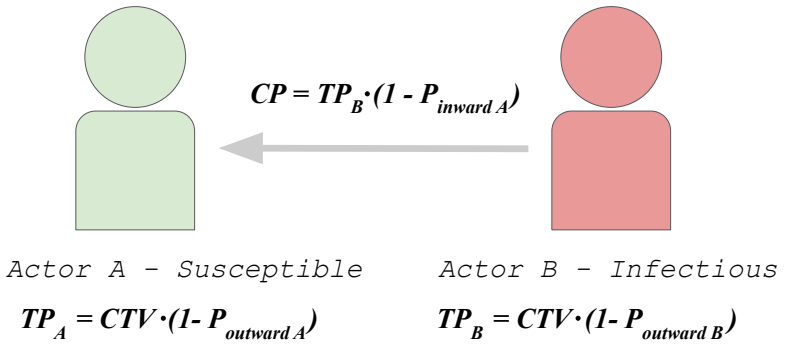

Figure 2.5: COVID-19 transmission process between two agents

2.2.7 Impact of Lockdown on Social Interactions

On March 8, 2020, the Italian government implemented strict lockdown measures to limit the virus's spread. These measures, collectively referred to as *Lockdown*, included the closure of non-essential services, bans on public gatherings, and restrictions on movement unless for urgent needs.

Social interactions dropped significantly during this period, as most people were limited to contact with household members. The average number of daily interactions for each age group during this phase is summarized in Table 2.1, assuming that most contacts occurred at home.

A notable exception was essential workers, who were allowed to continue work-

ing onsite. Their interaction patterns were different, as they were still exposed to contacts at work and in public transport. Table 2.4 lists the population percentage by age group who continued working outside the home during the lockdown [76].

Age Group	Essential Workers (%)
20-29	14.6%
30-39	25.4%
40-49	28.7%
50-59	22.7%
60+	8.5%

Table 2.4: Percentage of Italian population, by age, engaged in essential work during the lockdown

Use of Protective Equipment

During the lockdown, protective equipment, particularly masks, became widespread. According to [71], 83.81% of the Italian population wore masks during this period, with a margin of error of 2.23%. In the model, mask usage is assigned to individuals at the start of the simulation, with mask efficacy determined by the type of mask used.

Data from [72] informs the assumptions about mask effectiveness. The three mask types considered are cloth, surgical, and N95. Cloth masks offer 20-80% inward protection, while surgical masks range from 70-90%, and N95 masks provide over 95% protection. Outward protection varies similarly, with cloth masks offering 0-80% protection, surgical masks 50-90%, and N95 masks 70-100%. The overall average effectiveness of masks reduces transmission probability by 62-90% inwardly and 40-90% outwardly.

2.2.8 Experimentation in the Lombardy Region

To test the accuracy of the simulation model and its ability to replicate real-world social dynamics during the COVID-19 pandemic, it was applied to Lombardy, Italy, which experienced two major outbreaks in 2020. The primary focus was on the first

wave, which occurred from January to April 2020. The second wave, spanning August to December 2020, was later used to validate the model's robustness and ensure that its parameters remained reliable.

At the beginning of the simulation, all agents were assumed to be susceptible, meaning no infections had yet occurred. To mimic real conditions, a portion of the population was randomly selected to begin in the incubation phase. This selection was based on the actual number of positive cases reported in Lombardy between February 20 and February 29, ensuring that the model aligned with the real spread of the virus at the provincial level.

One of the core elements of the model is the transmission probability, which reflects the likelihood that an individual can spread the virus. This value needed to be carefully estimated to match real-world conditions. To achieve this, a random search was employed through the probability space, using real-world data as a guide. The focus was on the time period before the national lockdown, implemented on March 8, 2020, when mask-wearing and other protective measures were not yet widespread.

Transmission probability, ranging between 0 and 1, was adjusted to ensure that the simulation's output matched the real number of cases as closely as possible up to the lockdown date. After running several simulations and averaging the outcomes, the transmission probability 0.3 was determined to best fit the data. The relationship between transmission probability and the number of positive cases recorded on March 8 is shown in Figure 2.6.

As the pandemic progressed into the summer of 2020, the availability of testing increased, providing more accurate data. This enabled further validation of the model during the second wave. Incorporating these new data confirmed that the model could reliably simulate the spread of COVID-19, even as social behavior and testing capacities evolved.

Different simulation scenarios were also explored, with each scenario run multiple times (an average of 10 runs) to account for the inherent randomness in the process. Two key metrics were used in the analysis to assess how closely the simulation aligned with reality: the Pearson correlation coefficient and the Root Mean Square Error (RMSE). The Pearson correlation measures how closely two sets of data follow

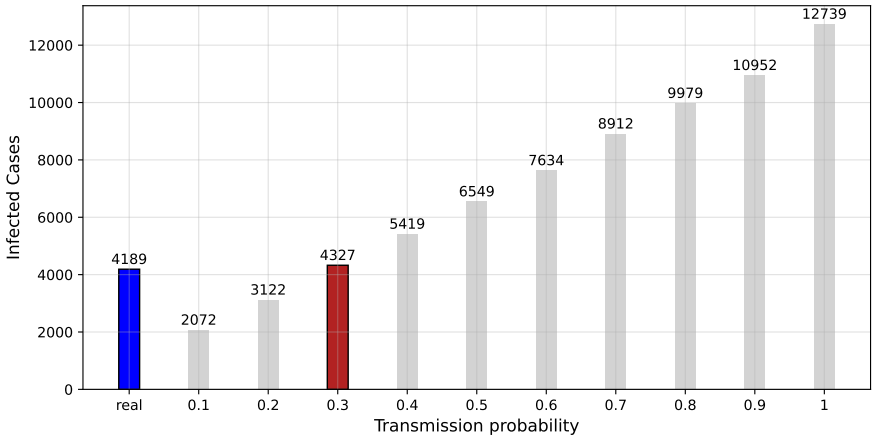

Figure 2.6: Distribution of positive cases on March 8, 2020, versus transmission rates.

the same trend, with values ranging from -1 to 1, where 1 indicates a perfect positive correlation. This case shows how well the simulated contagion curve reflects the real one. RMSE, on the other hand, measures the average difference between predicted values and actual observations, helping us gauge the model's overall accuracy.

The results from the first wave are presented In Figure 2.7, between January and April 2020, with a COVID-19 Transmission Probability (CTP) set at 0.3. The blue line represents the real number of positive cases, while the red line shows the simulation output. Until April 30, 2020, the Pearson correlation coefficient was 0.992, indicating a strong match between the two datasets. The RMSE was calculated at 38,818, meaning the simulator slightly overestimated the total number of positive cases by about 53,000 on that date.

These findings suggest that the model is highly able of replicating real-world contagion dynamics, particularly during the first wave of the pandemic. Although the simulation slightly overestimated the total number of positive cases, this may be attributed to early-stage limitations in testing or discrepancies in how cases were reported.

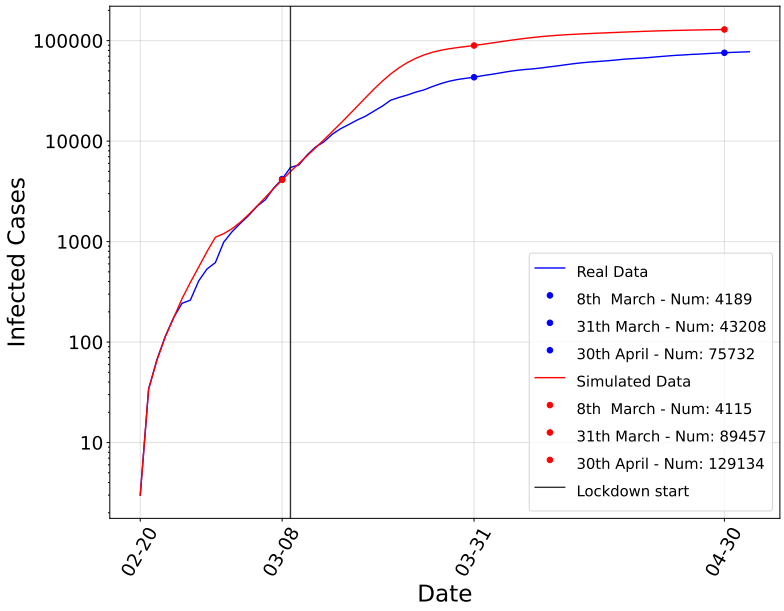

Figure 2.7: Simulation results compared to real data with starting assumptions (CTP=0.3) - Incremental representation of positive cases.

2.2.9 Parameters Fine-Tuning

The first wave of COVID-19 in Italy presented challenges in data accuracy, mainly due to the country's lack of preparedness and limited access to contagion tracking systems and widespread testing. As a result, the initial data collected from the outbreak were incomplete and unreliable. By the time the second wave hit in the autumn of 2020, better tools and testing mechanisms were available, allowing a more accurate analysis of the epidemiological data.

The previously developed simulation model was initially applied to the second wave, using data from September 15th to October 30th, 2020. The model was updated to reflect autumn conditions, but the simulated outcomes significantly underestimated the real infection data. As shown in Figure 2.8, the red curve (simulated data) diverged from the blue curve (actual data). The Pearson correlation coefficient for this

comparison was 0.988, with a root mean squared error (RMSE) of 27,415. This discrepancy indicated that the transmission probability parameter must be adjusted to fit the observed data better.

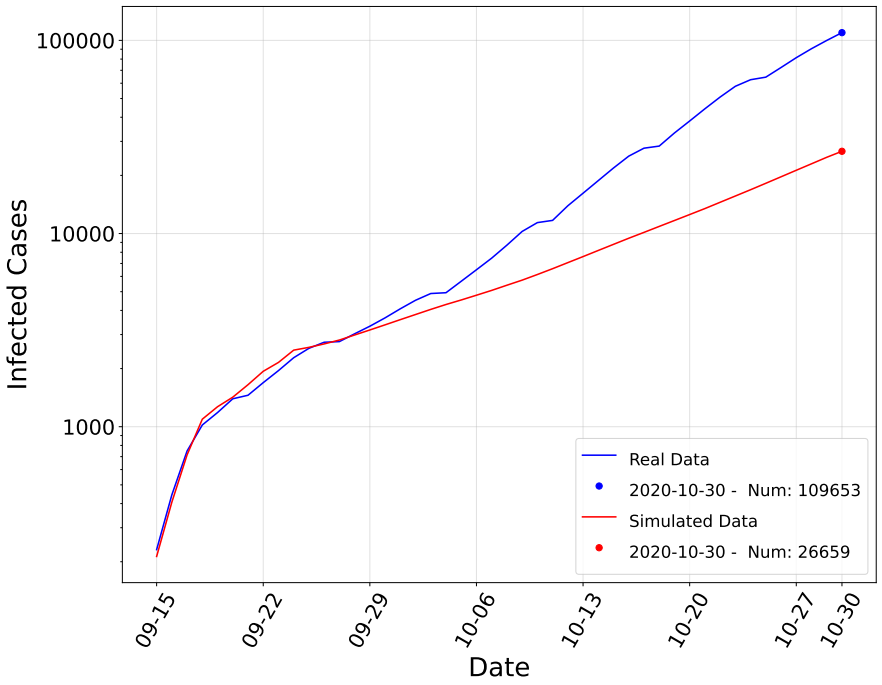

Figure 2.8: Simulation results with real data using an initial contagion transmission probability (CTP=0.3) during the autumn phase.

To address the estimation error, the transmission probability parameter was recalculated, focusing specifically on the autumn wave data. After an exhaustive search, a value of 0.53 was found to best capture the real progression of the contagion curve. This new simulation, shown in Figure 2.9, significantly improved the model's accuracy. The Pearson correlation coefficient increased to 0.996, and the RMSE decreased to 6,405, confirming a more precise alignment between the model and reality.

Based on these findings, the simulation for the spring 2020 wave was then rerun using the updated transmission probability, along with a comparison to national

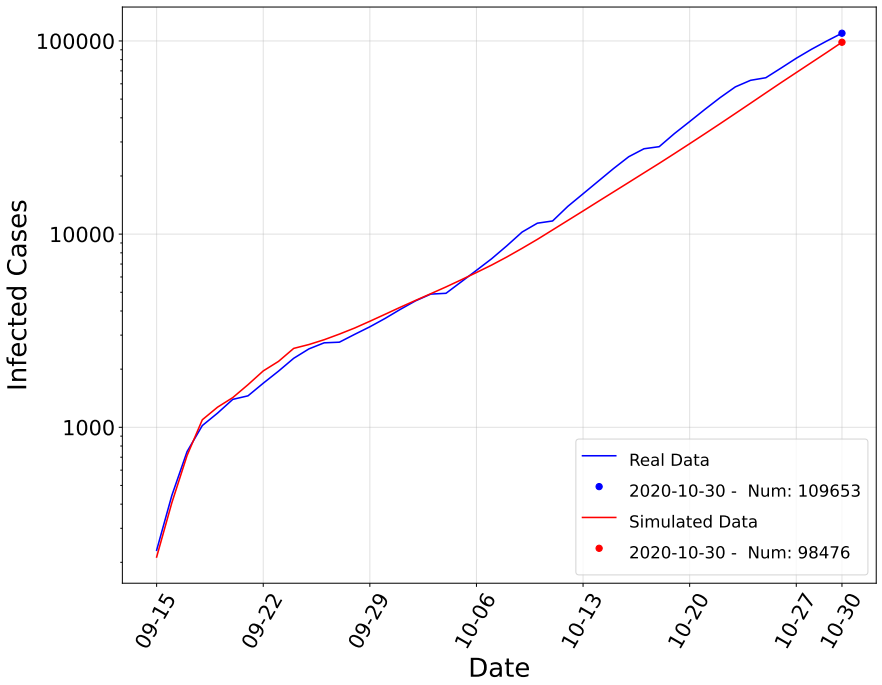

Figure 2.9: Simulation results with real data using a refined transmission probability (CTP=0.53) during the autumn wave.

serological screening data.

A different scenario was compared with the serological study by *Istat* [77]. By July 15, the actual number of COVID-19 cases in Lombardy was about 7.92 times higher than the cases reported by Civil Protection. The study found that 7.5% of Lombardy's population (about 754,500 people) had antibodies. On that same date, only 95,236 cases were confirmed by swabs, leading to the ratio:

$$\frac{754.500}{95.236} \approx 7.92$$

This suggests that spring data was significantly underestimated. Assuming this ratio held over time, actual infections can be retrospectively estimated from reported cases to serve as a benchmark for the mode.

Figure 2.10 shows this comparison, where the blue curve represents the actual data, the green curve reflects the serological projection, and the red curve represents the simulated data with a contagion probability of 0.53. The difference between the simulated and serological data on April 30th, 2020, was only 82,369 units. This close match strongly supports the accuracy of our simulation model, as seroprevalence analysis is much more reliable than case counts alone due to its inclusion of asymptomatic individuals. The Pearson correlation coefficient for comparing simulated and real data remained high at 0.996, with an RMSE of 249,529. Compared to serological data, the RMSE was 56,009, further validating the model's assumptions.

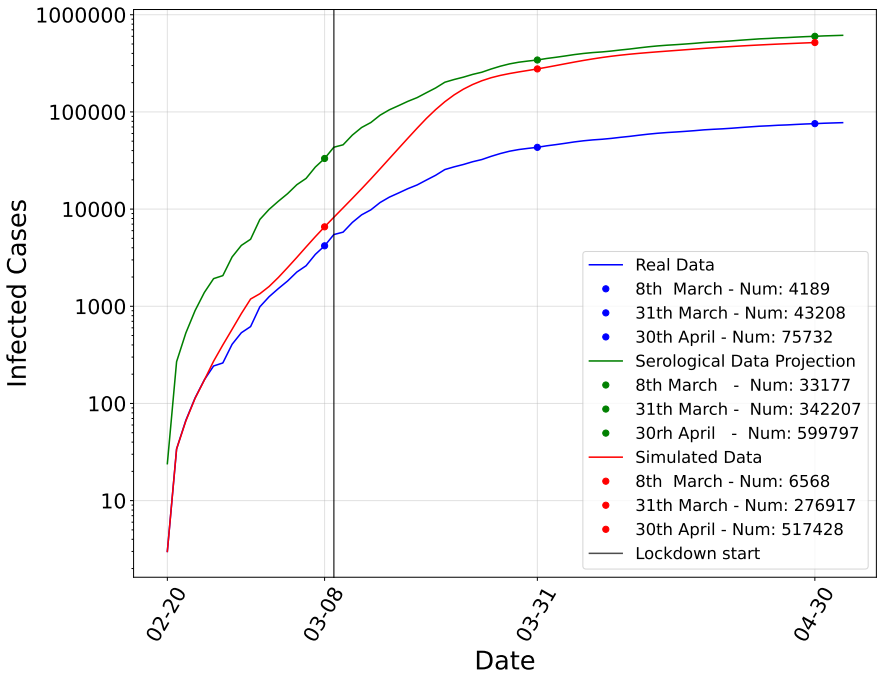

Figure 2.10: Simulation results with real data, using a transmission probability of 0.53 for the spring case, compared with serological data.

The lockdown measures imposed in Italy during the pandemic led to a dramatic reduction in social interactions, this was modeled in the simulation by adjusting so-

cial contact patterns. Essential workers were still allowed to move, but overall, the frequency of interactions declined significantly. The degree distribution of social contacts was examined at different time points during the lockdown to analyze these changes. Two candidate distributions, power-law, and log-normal, as well as log-normal and exponential, were evaluated by comparing the likelihood ratios with empirical data. As shown in Figure 2.11, the Complementary Cumulative Distribution Function (CCDF) trends during the lockdown predominantly followed a log-normal distribution, as indicated by consistently low p-values. This shift in the degree distribution highlights how lockdown measures reduced social interactions and gatherings, curtailing the role of super-spreader individuals and thereby influencing the overall epidemic dynamics.

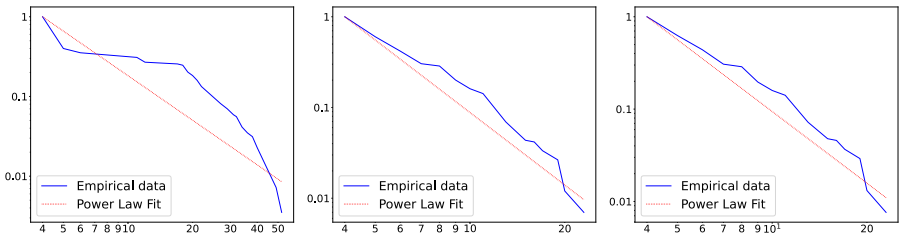

Figure 2.11: CCDF during the lockdown period, with empirical data in blue and power-law fit in red, showing changes in social interactions on March 8^{th}, March 31^{st}, and April 30^{th} (left to right).

To further assess the model's effectiveness in capturing real-world dynamics, it was compared against a null model that assumed random social interactions instead of a power-law distribution. In this alternative model, each agent met a fixed number of contacts (19 on average), regardless of individual social habits. The spring wave was simulated using various TP values, 0.1, 0.3, and 0.53, the latest equal to the value used in our primary model. The outcomes were averaged over five tests for each TP. Figure 2.12 and Table 2.5 summarizes the results, showing that the power-law based model provided a more reliable representation of COVID-19 spread, with superior RMSE and Pearson correlation values when compared to both real and serological data. This reinforces the importance of using a heterogeneous contact network to

accurately model epidemic dynamics.

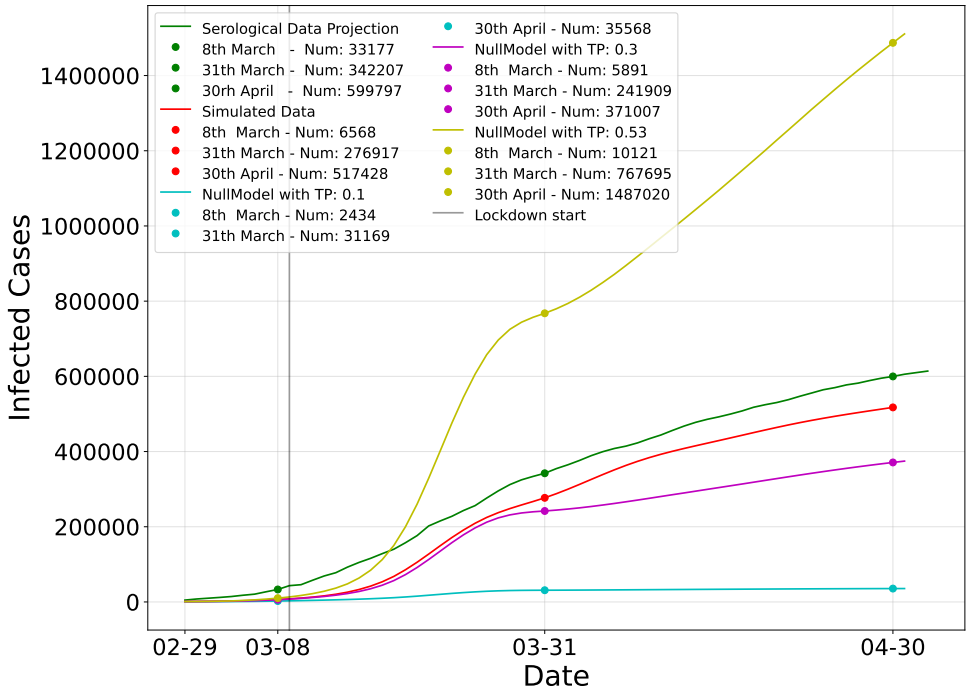

Figure 2.12: Simulation results using the null model (random interactions) for different transmission probabilities (0.1, 0.3, 0.53) during the spring case.

	RMSE	**Pearson Correlation**
Our model	56009	0.996
Null model with TP = 0.1	346040	0.956
Null model with TP = 0.3	139227	0.990
Null model with TP = 0.53	481493	0.994

Table 2.5: Comparison of our model against the null model in terms of RMSE and Pearson correlation for different transmission probabilities.

2.2.10 Comparison with State-of-the-Art SEIR Models

To validate the resulting COVID-19 model, it was compared against several SEIR models, as shown in Figure 2.13. Data from February 20th to 29th, 2020, were used in these comparisons. The following SEIR models were considered:

1. A traditional SEIR model that does not account for lockdowns, mask usage, or other restrictive measures.

2. The model by Riccio et al. [78], tailored for Lombardy, incorporates asymptomatic transmission, a variable R0 during lockdown, and policy impacts.

3. The model by Godio et al. [79] uses a SEIR approach optimized by a Swarm Intelligence algorithm and incorporates an exponentially decreasing R0 to account for mask adoption, lockdowns, and deaths.

The results for March-April 2021, Table 2.6, show that the model produces more accurate predictions of COVID-19 spread in Lombardy (based on serological data), outperforming the SEIR models in terms of both RMSE and Pearson correlation.

	RMSE	**Pearson Correlation**
Our model - Serological	56009	0.996
Riccio et al. - Serological	270060	0.859
Godio et al. - Serological	318034	0.713
SEIR - Serological	269197	0.769

Table 2.6: Comparison with SEIR models in terms of RMSE and Pearson Correlation

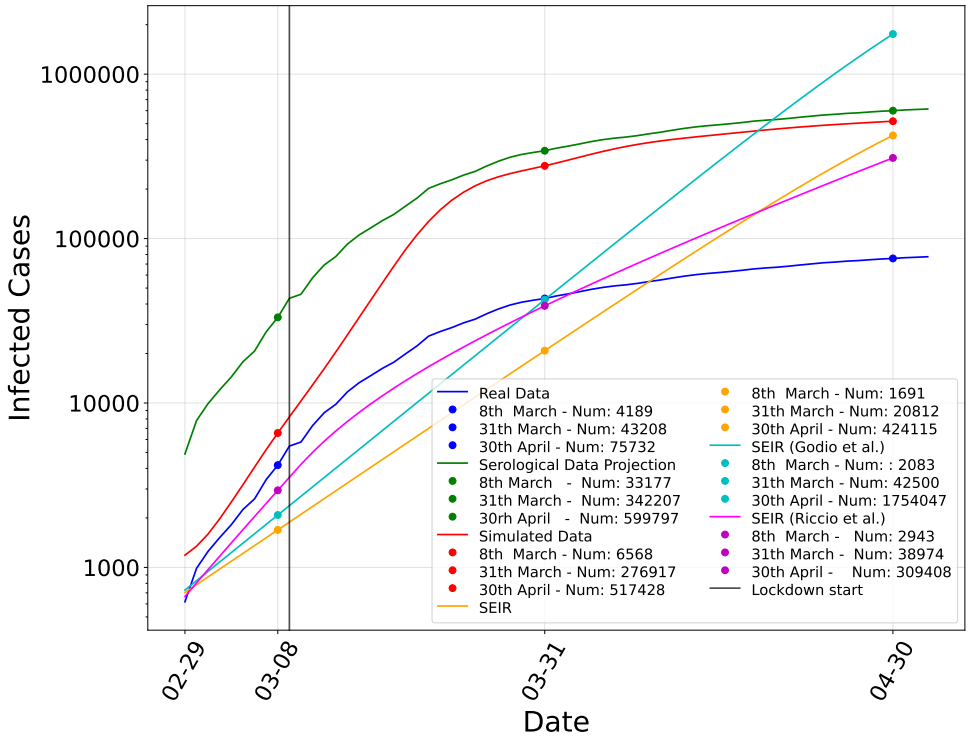

Figure 2.13: Comparison of our model, real, serological, and SEIR models. Spring case - Incremental positives representation

2.3 Emilia-Romagna Use Case: Refining and Validating the Contact Network Model

In the continued effort to enhance the accuracy of COVID-19 epidemic simulations, a more thorough analysis of contact networks was conducted. After applying the fine-grained model to Lombardy, a different regional context was sought to validate and refine the model further. With its unique demographic and economic features, Emilia-Romagna presented an ideal case for this purpose. The previous model had shown promising results, but the varied urban, rural, and industrial dynamics of Emilia-Romagna provided an opportunity to validate its robustness across different

scenarios.

In this section, I explore how the refined simulation framework, ABMs and HPC, was adapted for the Emilia-Romagna case, and how its results compared with the Lombardy ones.

2.3.1 Enhancing the Contact Network Model

The contact network model is critical in simulating the spread of an airborne virus like COVID-19. To capture various social interactions, the existing model used in Lombardy was built upon for further analysis by introducing more granular distinctions in the types and frequencies of contacts.

Emilia-Romagna, unlike Lombardy, is a mix of urban, industrial, and rural zones, each with its own patterns of human movement and interaction. Therefore, the social network structure in this region is more complex. Using mobility data and adjusting for sociability rates across different population centers, the model was adjusted to accurately reflect the true nature of social interactions across the region.

The average number of daily contacts was adjusted based on whether individuals lived in urban or rural settings. This distinction allowed for more precise modeling of infection spread in less densely populated areas, where social interactions are fewer but potentially longer.

2.3.2 Commuting Patterns and Mobility Modeling

Emilia-Romagna's economy is highly industrialized, and many residents commute daily between cities and rural areas. This created a distinct challenge in modeling the region's infection dynamics, as commuting plays a significant role in the spread of airborne diseases. In order to accurately reflect this, a commuting model based on gravity models and mobility networks was introduced.

For high school students, university students, and workers, there is the possibility that the employment place is located in a different town than that of residence, or even in a different province. Commuting is very common nowadays and plays an important role in the infection spreading even in the most isolated municipalities. Therefore,

it is essential to model this phenomenon in a way that is as closely comparable to reality as possible. Commuting models commonly rely on the effectiveness of two parameters:

- The grain: the finer the grain of the model, the more accurate the commuting model is.
- The economic quotient: the most effective models take into account the economic quotient of every zone. Moreover, a municipality with a high number of businesses attracts more workers than another with few job opportunities.

Following extensive analysis, the decision was made to use the average results from the gravitational model outlined in [80] and the radial model presented in [81]. This approach was chosen because the gravitational model offers better commuter distribution across municipalities near the departure point. On the other hand, the radial model provides a more accurate estimate of the total number of individuals moving from a specific municipality. However, it tends to concentrate movement towards the nearest towns. The gravitational model relates the population of the origin and destination municipalities with the Euclidean distance between them:

$$c_{i,j} = \theta \frac{N_i^{\tau f} N_j^{\tau t}}{d_{i,j}^{\rho}} \tag{1}$$

where:

1. $c_{i,j}$ is the probability that an individual living in i works or studies in municipality j
2. N_i, N_j number of individuals living in municipality i (or j)
3. θ proportional constant equal to 0.0005
4. τf inhabitants damping constant of i equal to 0.28
5. τt inhabitants damping constant of j which changes according to the number of people living in j:
   (a) 0.65 if the number of inhabitants is greater than 150,000

(b) 0.66 if the number of inhabitants is between 5000 and 150,000

(c) 0.78 if the number of inhabitants is less than 5000

6. $d_{i,j}$ distance between the municipalities i and j

7. ρ constant amplifying the dependence from distance which changes according to the number of people living in j:

   (a) 3.05 if the number of inhabitants is greater than 150,000

   (b) 2.95 if the number of inhabitants is between 5000 and 150,000

   (c) 2.5 0.78 if the number of inhabitants is less than 5000

Due to the absence of data on an economic quotient, it was assumed that municipalities with larger populations also have greater economic attraction. As a result, the constants were adjusted based on the population size of each municipality. In contrast, the radial model establishes a relationship between the population of the departure municipality, the population of the destination municipality, and the number of people residing within a circle whose radius corresponds to the distance between the two municipalities. Therefore:

$$p_{i,j} = p_i \frac{N_i N_j}{(N_i + S_{i,j})(N_i + N_j + S_{i,j})} \tag{2}$$

where:

1. $p_{i,j}$ is the probability that an individual living in i works or studies in municipality j

2. p_i initial commuting probability of municipality i. This parameter can assume three different values according to the number of people who live in i:

   (a) if the number of inhabitants is greater than 150,000

   (b) 0.3 if the number of inhabitants is between 5000 and 150,000

   (c) 0.4 if the number of inhabitants is less than 5000

3. N_i, N_j number of individuals living in municipality i (or j)

4. $S_{i,j}$ population that lives within the circle with radius equal to the distance between i and j minus the population living in i and j

The values of the constants were determined by applying and comparing the model to the Italian context. Specifically, the constants were identified using data from the Emilia Romagna region [82], which was then combined with features extracted from [80]. Figure 2.14 illustrates the differences between the gravity and radial models. In the gravity model, the Euclidean distance alone is sufficient to estimate the commuting probability between Albinea and Reggio nell'Emilia. In the radial model, however, it is also necessary to account for the population within the area defined by the distance between the two municipalities.

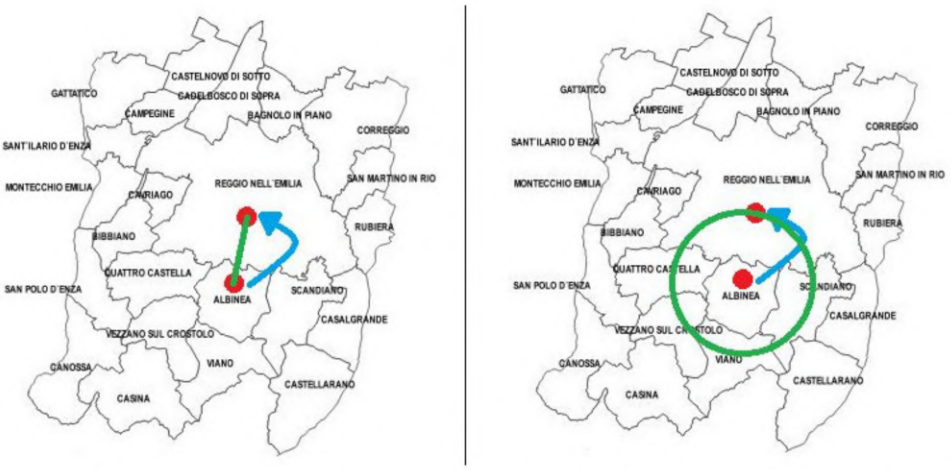

Figure 2.14: Image representing the difference between the gravitational model (left) and the radial model (right). In this example, the municipality of Albinea (departure) and Reggio nell'Emilia (arrival) are considered.

2.3.3 Socio-Demographic Model for Reggio Emilia Province

To realize a working model for the Reggio Emilia province, data on contagion at the municipal level was collected [83], along with additional data for the Emilia Ro-

magna region from [84]. Unfortunately, it was not possible to gather municipality-level contagion data for the entire Emilia-Romagna region. Therefore, to determine the initial condition, Reggio Emilia province's distribution was analyzed, dividing the municipalities into different classes according to the number of inhabitants. This class format was then extended to the whole region. The geographical coordinates, mandatory for calculating the commuting model distance, were retrieved from [85], and the population size and distribution were provided by ISTAT [86].

To correctly tune the gravitation models and radial models, it was necessary to take into account the commuting values provided by the region [87]. The use case considers the second outbreak that affected the entire Italian territory. Therefore, the simulation period concerns the trend of infections between 1 September 2020 and 15 December 2020. Finally, to validate our model, it was compared with the official data provided by the Italian Government (Protezione Civile) and the regional government of Emilia-Romagna.

Family groups and occupations (work or student) were introduced for the individuals involved in the simulation, leading to the definition of a new socio-demographic model. The municipalities, with their geographical coordinates and the agents representing the population, were created following the data and probability distributions provided by the census. In the model, there are nine different family types, characterized by a different composition and number of members:

1. Single with children
2. Single without children
3. Single with children plus another adult
4. Couple without children
5. Couple without children plus another adult
6. Couple with children
7. Couple with children plus another adult
8. Adults living together

9. Family groups (with at least one child)

The term "other adult" refers to a person who is not strictly part of the family group but lives in the same habitation. Each type of family has a frequency of appearance, and the number of members may vary depending on a distribution probability, shown in Figure 2.15.

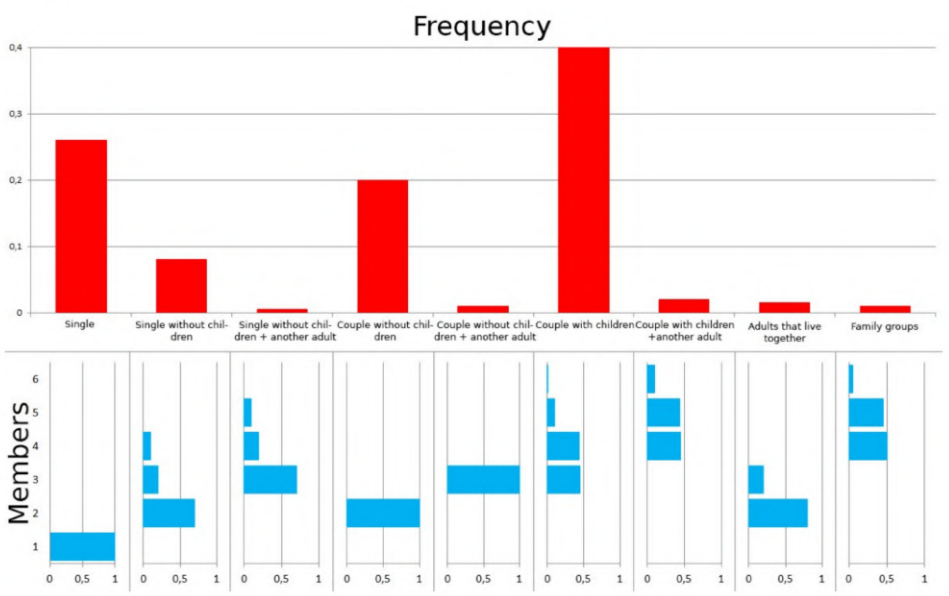

Figure 2.15: Family type distribution chart with the relative number of members.

Three fundamental constraints for building a consistent family group were introduced:

1. Any family group must contain at least one adult.
2. The age of each child must be between 18 and 43 years less than the younger parent.
3. The age difference between a couple is less than or equal to 15 years, and both must be adults.

Each individual was assigned an occupation based on age, and probability distributions were applied based on the attendance rate. Table 2.7 shows the attendance rates and class sizes for different school categories, as well as the employment rates for different age groups and the company sizes in the simulation.

In addition, each type of company exhibits a distinct frequency of appearance, as illustrated in Figure 2.16.

Category	Attendance Rate/Employment Rate	Class/Company Size
School Categories		
Kindergarten	90%	40 children
Preschool	90%	20 children
Elementary School	100%	19 children
Middle School	100%	21 students
High School	92%	21 students
University	31%	34 students
Employment Rates by Age Group		
15–19 years	8%	—
20–26 years	30%	—
27–34 years	62.5%	—
35–54 years	73.5%	—
55–70 years	54.3%	—
Company Sizes		
Very small company	—	up to 5 employees
Small company	—	up to 9 employees
Small-medium company	—	up to 19 employees
Medium company	—	up to 49 employees
Medium-large company	—	up to 99 employees
Large company	—	up to 249 employees
Very large company	—	over 250 employees

Table 2.7: Occupation, school categories, and employment rates.

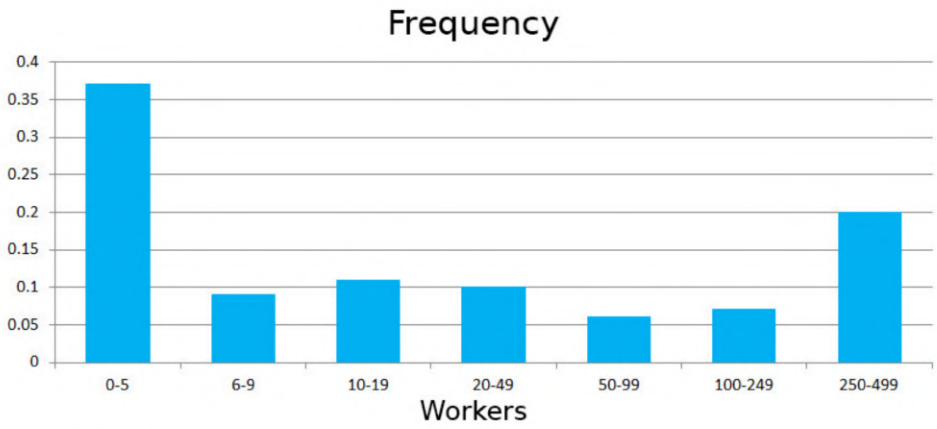

Figure 2.16: Distribution of work classes.

2.3.4 Adaptive Lockdown Strategies

In the Emilia Romagna model, six distinct levels of restrictions were implemented to accurately reflect the division into zones (or "bands") imposed by the Italian government during the simulation period. These restrictions evolved to simulate the changes in public policy:

1. White (no restrictions): The baseline scenario with no limitations on movement or social activities.

2. White (10/18 to 10/24): This phase reflects the initial restrictions introduced on 18 October 2020, where no restrictions were placed on work, but high schools and universities shifted to 50% in-person attendance. Social interactions decreased by 40%, with measures like the early closure of bars (9 p.m.) and restaurants (midnight).

3. White (10/25 to 11/05): Additional restrictions were introduced here, still allowing workplaces to function, but schools and universities remained at 50% attendance. Activities such as gyms, theaters, and cinemas were closed, and restaurants shut down after 6 p.m. Citizens were also encouraged to avoid un-

necessary travel. This resulted in a 50% reduction in social interactions.

4. Yellow: High schools and universities were fully closed (100% distance learning), and a curfew was introduced at 10 p.m. Cultural sites like museums and libraries were shut down. These restrictions led to a 60% reduction in social interactions.

5. Orange: In addition to the yellow zone restrictions, travel between municipalities was prohibited except for work, and dining establishments (except for take-out) were closed. Smart working was highly encouraged. This level resulted in a 70% reduction in social interactions, which were now limited to within municipalities.

6. Red: The strictest level, where even movement within municipalities was restricted. Daily interactions were reduced by 80%, with non-essential social interactions essentially eliminated.

To accurately model the spread of infection, it was necessary to fine-tune the TP in the simulator, a key parameter in the *ActoDemic* framework. An empirical method was employed, running ten simulations for each TP value and adjusting it via a random search through the probability space. The goal was to align the simulation results with real-world infection data for the Reggio Emilia province, Figure 2.17, and the broader Emilia Romagna region, Figure 2.18.

For the Reggio Emilia province, the optimal TP value was determined to be 0.285. However, when applying this value to the entire Emilia Romagna region, it proved too high. This discrepancy was due to the influence of inter-provincial commuting, which affected contagion dynamics across neighboring provinces. After further analysis, 0.25 was identified as the most suitable TP value for the entire region.

2.3.5 Results and Model Validation

After determining optimal transmission probabilities (TP) for the two scenarios, the models were re-simulated to evaluate the impact of the implemented restrictions. For the Reggio Emilia province, with a TP of 0.285, the daily infection trend was

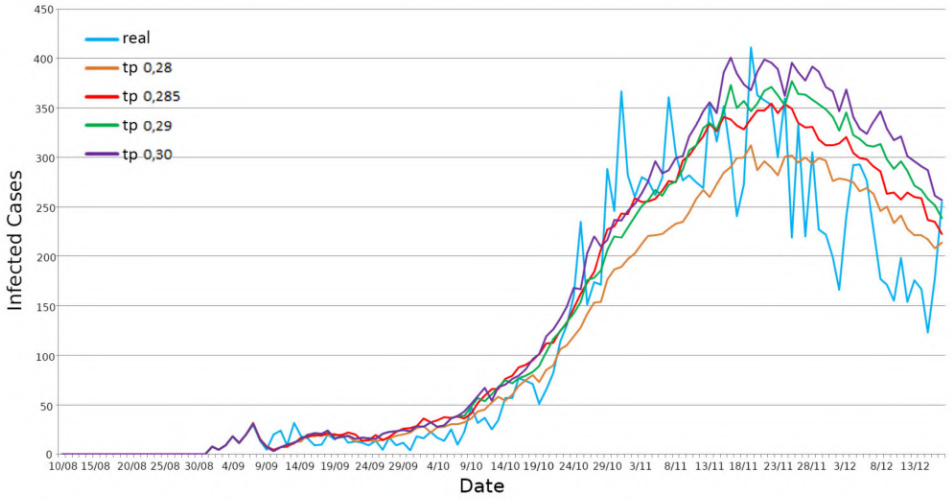

Figure 2.17: Graph of daily infected persons with different TP values for Reggio Emilia province.

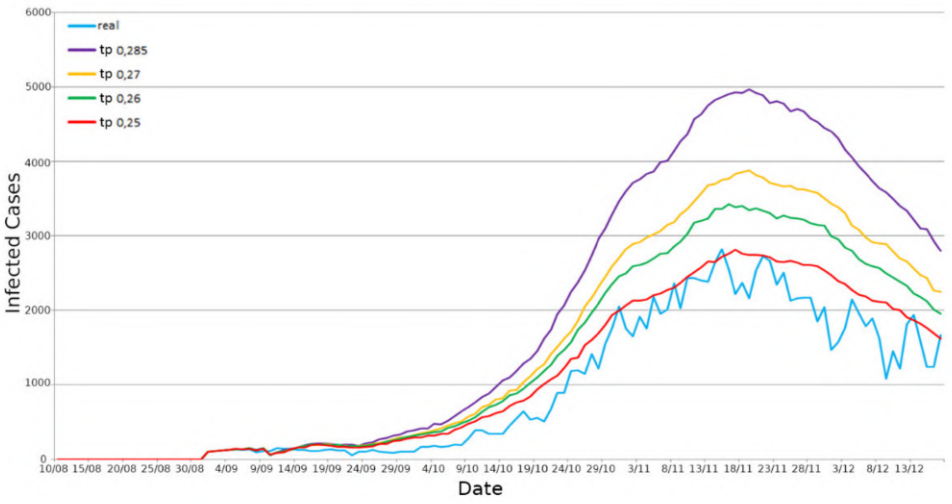

Figure 2.18: Graph of daily infected persons with different TP values for the Emilia Romagna region.

analyzed, Figure 2.17. This simulation yielded an R^2 value of 0.854, a Pearson correlation of 0.941, and a root mean square error (RMSE) of 48.32, Figure 2.19. The simulated curve closely mirrored the actual infection trend, with some discrepancies attributed to variations in daily COVID-19 testing rates. To address data noise, a cumulative curve was created, resulting in improved metrics: an R^2 of 0.986, a Pearson correlation of 0.9988, and an RMSE of 608.88, Figure 2.20.

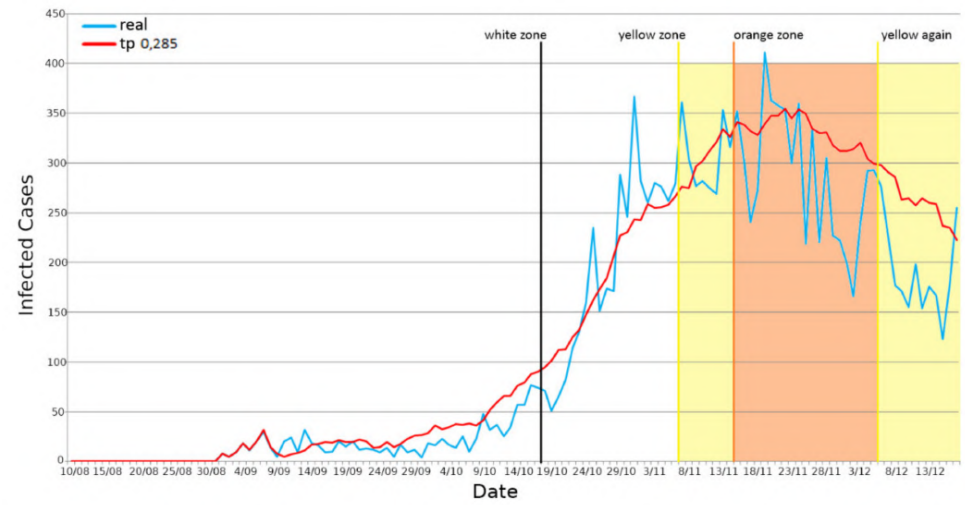

Figure 2.19: Positives' daily representation – Reggio Emilia province, Emilia Romagna region.

The analysis was extended to the Emilia Romagna region using a TP of 0.25. Both daily and cumulative infection trends were assessed. The results for daily infections, shown in Figure 2.21, produced an R^2 of 0.881, a Pearson correlation of 0.977, and an RMSE of 319.08. The cumulative infection data displayed in Figure 2.22 yielded even stronger results, with an R^2 of 0.922, a Pearson correlation of 0.9993, and an RMSE of 10,730.11. These outcomes indicate that our simulator is reliable, successfully modeling agent movements in both provincial and regional contexts. Consequently, it can effectively assess the impact of restrictions and predict their efficacy in curbing the epidemiological curve.

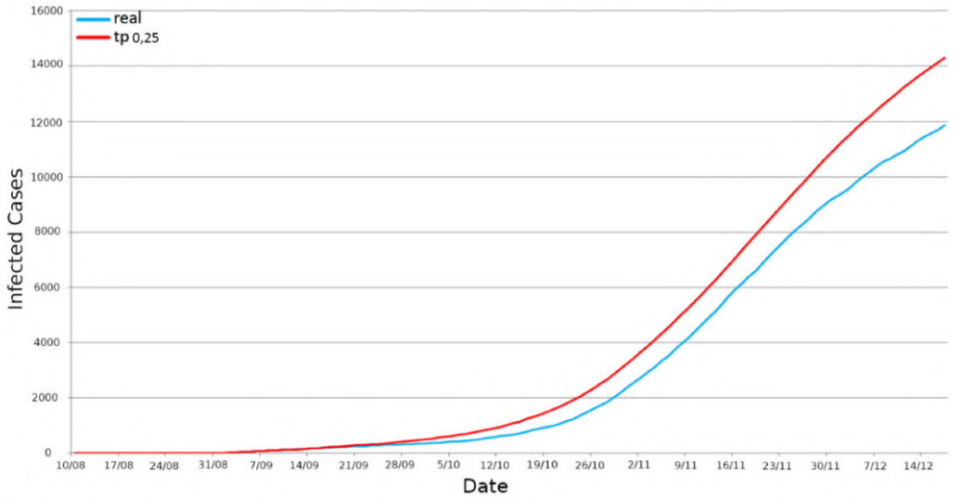

Figure 2.20: Positives' incremental representation – Reggio Emilia province, Emilia Romagna region.

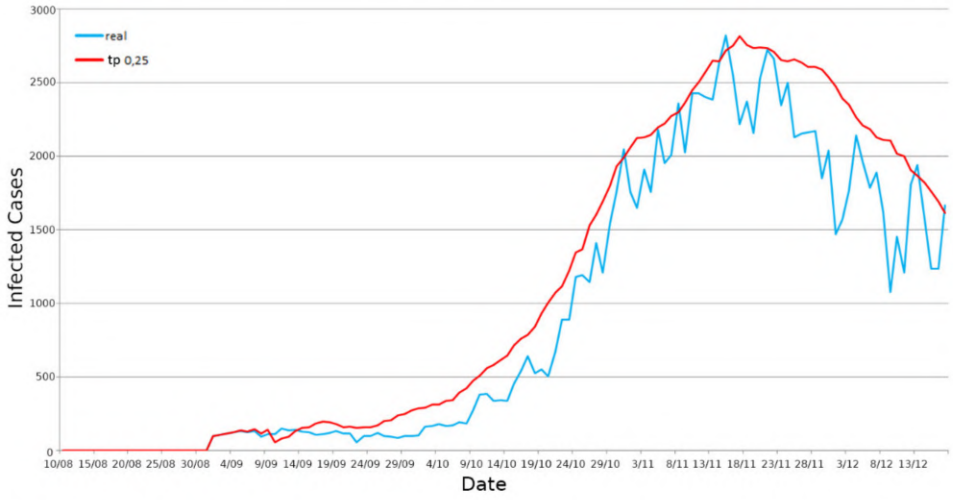

Figure 2.21: Positives' daily representation – Emilia Romagna region.

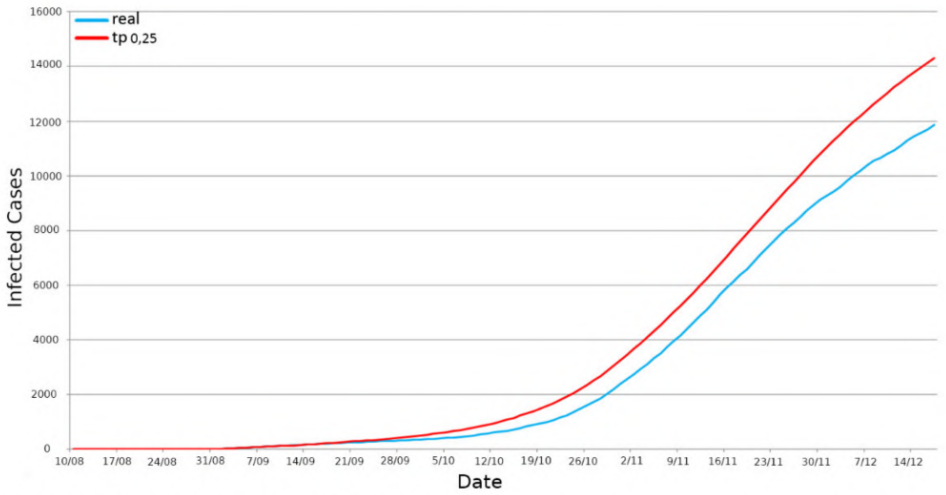

Figure 2.22: Positives' incremental representation – Emilia Romagna region.

Commuting model in Lombardy region

For the further validation of our model, the use case previously implemented in the research work was employed. In the case of the Lombardy region, where high-resolution municipality data was unavailable, validated features from the Emilia Romagna region were scaled and applied across its ten provinces. This approach reinforced the robustness of our methodology and ensured that the model could be applied to a variety of regional contexts in conditions of poor localized data. The analysis focused on the initial phase of the pandemic, spanning from February 20 to April 30, 2020. The implemented commuting algorithm treated each province as a major urban center. Notably, real data during this period were likely underestimated due to testing limitations, with actual cases possibly eight times higher, as confirmed by serological analyses conducted by Italian authorities in July 2020 [77]. The simulated incremental contagion curve was compared with a serological projection representing untracked COVID-19 cases, demonstrating close alignment, Figure 2.23. Qualitative comparisons of our simulation results, highlighted in Table 2.8, affirm the validity of

the commuting-based model against serological data projections.

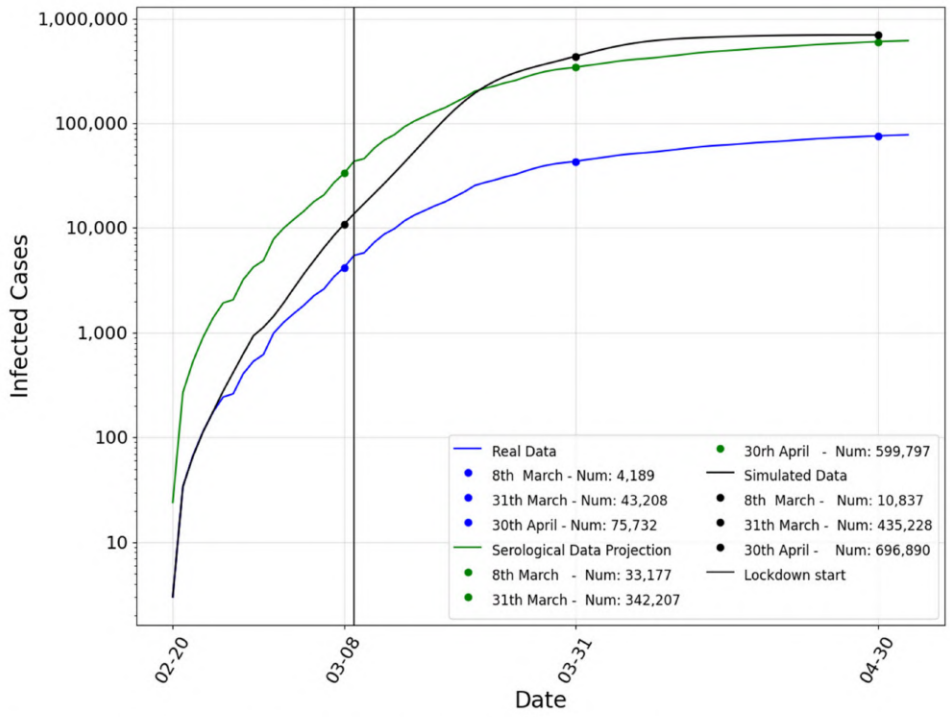

Figure 2.23: Positives' daily representation – Lombardy region.

	RMSE	Pearson Correlation
Simulated Data – Real Data	398,042	0.9903

Table 2.8: Pearson correlation and root mean squared error (RMSE) of our model concerning the serological data projection.

This analysis reveals that the simulation framework, grounded in commuting mechanisms, produces plausible and robust results, confirming the effectiveness of the implemented agent-based modeling system (ABMS).

2.3.6 Futher Analysis

Further tests and simulations were conducted to validate the model. The high-resolution model integrates various factors and parameters influencing the infection spread process. Specifically, the following aspects were examined:

1. Contagion susceptibility by age,
2. The protection offered by wearable protective devices,
3. Quarantine mechanisms.

Additionally, these parameters were assessed to determine their impact on the spread of the infection. A simplified model was provided that does not consider these factors; instead, it bases the infection process solely on transmission probability without any additional dampening effects. The results in 2.24 illustrate different infection trends corresponding to various transmission probability (TP) values. Moreover, the data in 2.3.6 confirm that the model's parameters provide sufficient contagion dampening, validating our infection modeling approach.

	RMSE	**Pearson Correlation**
Simulated Data – Serological	105,343	0.9903
Null model with TP = 0.2 – Serological	1,301,649	0.8674
Null model with TP = 0.4 – Serological	4,662,723	0.9456
Null model with TP = 0.6 – Serological	5,840,197	0.9805
Null model with TP = 0.8 – Serological	6,422,819	0.9763
Null model with TP = 1 – Serological	6,759,554	0.9616

Table 2.9: Comparison of our model to the null models in terms of root mean squared error (RMSE) and Pearson correlation.

Moreover, Further studies have been performed to improve analysis. The results were compared with those from other SEIR models to validate our model, as shown in 2.25. Three different SEIR models were considered:

- SEIR 1 a basic system dynamics SEIR model,

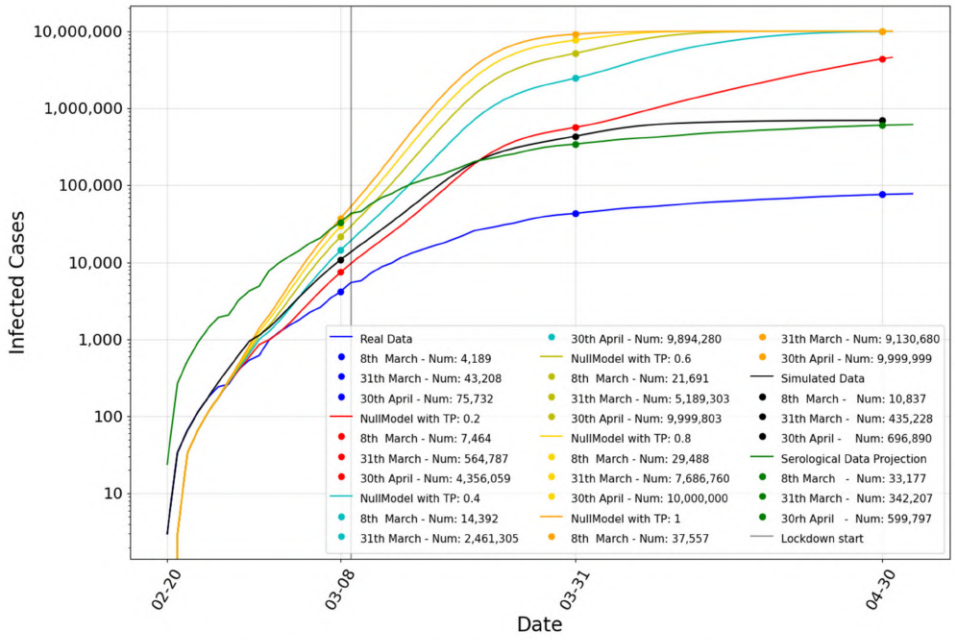

Figure 2.24: Simulation results without damping parameters with different transmission probabilities (0.2, 0.4, 0.6, 0.8, 1) (incremental representation – y-axis scale is logarithmic).

- SEIR 2 [79], which accounts for the transmission rate of asymptomatic individuals and a piecewise exponentially decreasing R_0,

- SEIR 3 [78], which employs computational swarm intelligence to optimize model parameters for the early stage of the pandemic in Italy.

A numerical analysis was also provided, comparing these models to the realized model in Table 2.10. The chart and the analysis demonstrate that the model yields more accurate predictions of infection spread regarding RMSE and Pearson correlation.

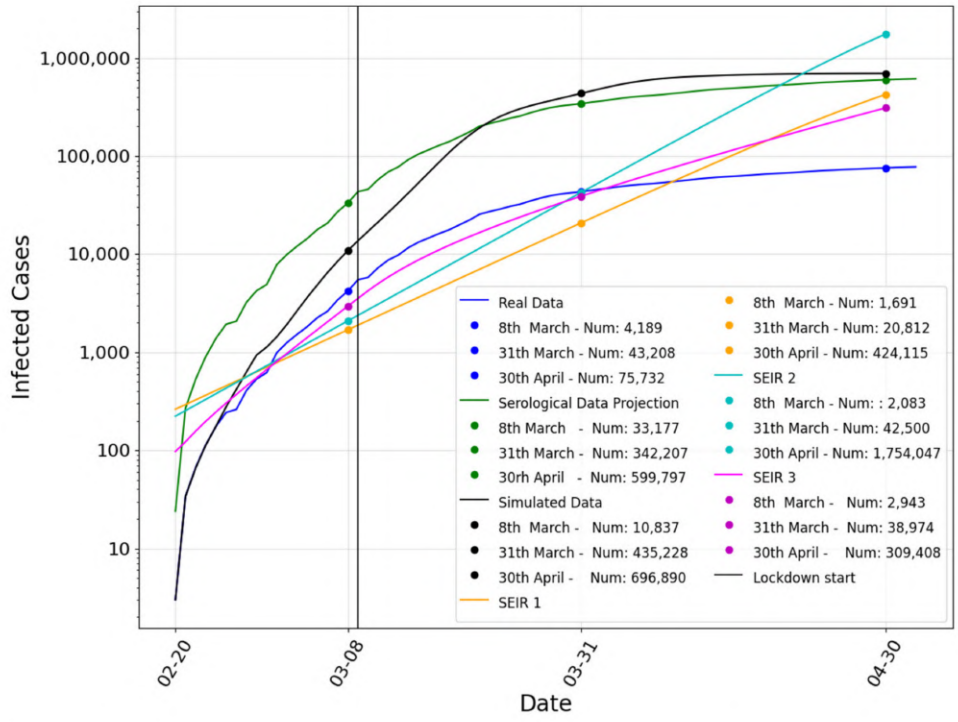

Figure 2.25: Comparison among our model, the serological data, the real data, and the SEIR models (incremental representation – y-axis scale is logarithmic).

	RMSE	**Pearson Correlation**
Simulated Data – Serological	105,343	0.990
SEIR 1 – Serological	269,197	0.769
SEIR 2 – Serological	270,060	0.859
SEIR 3 – Serological	318,034	0.713

Table 2.10: Comparison of our model to the SEIR models regarding root mean squared error (RMSE) and Pearson correlation.

2.4 Final Remarks

I introduced *ActoDemic*, a robust, modular framework that integrates a fine-grained simulation model with a large-scale epidemic scenario. The framework leverages an ABMS that can run efficiently on HPC facilities, which allows for the simulation of millions of concurrent agents. This capability was validated through the simulation of the COVID-19 outbreak in Lombardy, Italy, during 2020. The model incorporated key socio-demographic data, lockdown policies, protective measures, and a social interaction network. The results of the simulation closely matched real-world data, including the Italian seroprevalence studies conducted in 2020, demonstrating that *ActoDemic* is highly effective in replicating complex epidemiological phenomena. Notably, it was able to predict the number of infectious individuals, aligning with real estimates from the population of Lombardy.

During this simulation, *ActoDemic* successfully managed to handle up to 10 million agents, requiring substantial computational resources, specifically 600-800 GB of RAM and 32 CPUs, with an average execution time of 6 hours per simulation. This performance was enabled by the HPC facilities at the University of Parma, which were critical for scaling up the model to a regional level. The framework's modular architecture also made it user-friendly and customizable, allowing for the inclusion of new features as needed.

Building upon the success in Lombardy, the application of *ActoDemic* was expanded to Emilia-Romagna, another region in Italy. In this case, a novel commuting model was presented to simulate the effects of changing social behaviors on the spread of COVID-19. This model was essential in accurately reflecting movement patterns and interactions specific to the region. The simulation results from Emilia-Romagna were similarly promising, achieving an RMSE (Root Mean Square Error) of 10,730 during the autumn of 2020, compared to an RMSE of 56,009 for Lombardy in the spring of 2020. These findings indicate that our multi-agent approach, combined with detailed social modeling, provides significant insight into epidemic dynamics and highlights how different regions' behaviors can influence the progression of a pandemic.

The experiment further proved that the agent-based model in *ActoDemic* operates at a low abstraction level, effectively simulating agents' behaviors and interactions. Specifically, agent scheduling is message-driven, ensuring that agent behavior does not depend on execution order during each simulation step, making the framework scalable and efficient for large-scale simulations. Using "remote proxy" references and message aggregation provided by *ActoDeS* reduced communication costs and simplified the development process.

Future developments will focus on enhancing the framework's scalability and customization, with the goal of supporting even larger agent populations while requiring fewer resources. Additionally, exploring alternative message exchange paradigms and multi-agent systems could lead to significant performance improvements. There are plans to further extend the applicability of the model of *ActoDemic* to other domains, such as financial modeling, troll detection, and temporal graph-based tasks while continuing to refine the simulation models. This work will not only enhance the framework's capabilities for epidemic modeling but also open new opportunities for tackling complex real-world challenges.

Chapter 3

Proposed Cyber-Physical Approach in Medicine

The second part of this thesis delves into the application of CPS in the medical domain, explaining how CPS can contribute to and build upon advanced digital technologies for more effective management of a healthcare environment, especially in a surgical environment. Standing at the intersection of technology and healthcare, the rate at which things are changing is fast-moving, and the aim of this study is to show precisely how CPS might be applied to reduce inefficiencies and concurrently enhance the quality of patient care and improve the utilization of resources within hospital operating blocks (OBs).

This current research path results from a collaboration between the Department of Engineering and Architecture and the Department of Medicine and Surgery of the University of Parma, born out of the mutual need to improve management in operating theaters. Both departments felt the need to overcome traditional ways of managing surgical schedules, tracking patient movements, and allocating resources since these traditional methods were becoming increasingly insufficient for today's healthcare environments. This concept emerged upon the recognition that any cyber-physical solution must leverage the technical expertise of engineers to align with that of medical professionals.

In this chapter, DT model is introduced, with a detailed explanation of its components and how it functions as a virtual replica of the physical operating environment. The (Bluetooth Low Energy) BLE [88] IoT tracking system, which serves as a critical part of the physical layer of the DT, will then be illustrated. This integration allows for real-time data collection and monitoring, enhancing the operational capabilities of the OB.

The following sections will outline:

- An overview of the DT architecture and its key components, including the physical, digital, and communication layers.
- The role of the BLE IoT tracking system as part of the physical layer, highlighting its functionality and benefits.
- The integration of the IoT system into the broader DT framework illustrates how real-time data enhances predictive analytics and operational insights.
- The rationale behind implementing this system in the surgical environment and the expected improvements in patient care and resource management.
- The results observed from the implementation
- Future directions for expanding the DT model and integrating additional data sources for further optimization.

By focusing on the interplay between IoT technologies and the DT paradigm, I aim to demonstrate how this cyber-physical approach can revolutionize the management of operating rooms (OR), ultimately leading to improved healthcare outcomes and more efficient hospital operations.

3.1 The Key Role of Operating Rooms in Hospital Care

ORs play a major role in both hospital profits and expenses [89]. Around 60% of patients admitted to the hospital undergo procedures in the OR [90]. The efficiency of surgical scheduling is crucial, as it starts with estimating the duration of daily surgeries. When surgeries consistently take longer than expected, OR overuse leads to

costly overtime pay and dissatisfaction among staff. On the other hand, if surgeries finish earlier than predicted, it results in OR underuse, causing staff downtime and increasing costs by up to 60% [91, 92, 93]. Additionally, decisions about how many surgeries are scheduled in a day or week significantly impact the use of resources and patient flow. Balancing these decisions with tools like optimization algorithms and simulation models can help improve efficiency in healthcare settings [94]. Simulation tools are particularly helpful for evaluating processes, testing their efficiency, and predicting how revised or optimized processes might perform before implementing them in real life. They can also help plan for unexpected scenarios [95]. In most cases, the surgeon estimates the duration of a procedure when reserving an OR slot. Still, studies show that surgeons underestimate case times 42% of the time and overestimate them 32% of the time [96]. A common alternative is to use electronic health records (EHRs) to estimate case durations based on historical data, which tends to be more accurate [97]. However, EHR data usually only provide general estimates for the average patient and don't account for specific patient characteristics, such as age, BMI, allergies, or existing health conditions [98].

With the rise of big data, new opportunities exist to make patient-specific predictions using ML and deep learning (DL) techniques. Supervised learning models can detect patterns in large datasets by training on past data and then generalizing for new, unseen cases. This capability has sparked a growing interest in ML for clinical and organizational healthcare uses. However, challenges remain, particularly around ensuring accurate and clean data inputs, which is crucial for model success [99].

In light of these challenges, the research aims to develop a comprehensive technological and organizational model that leverages data from ORs to optimize the management of the entire OB.

3.2 Use Case: Ospedale Maggiore di Parma

3.2.1 Background

The proposed approach was implemented in the OB of the Maggiore Hospital in Parma. Before adopting the proposed solution, the OB had various challenges and

problems that impacted its efficiency and effectiveness. These included inefficient resource utilization, a lack of real-time tracking, difficulty coordinating medical staff and procedures, and limited access to historical data for performance analysis and improvement. As a result, delays, inefficiencies, and poor patient care were possible. Addressing these issues requires a comprehensive solution that could optimize operations, enhance communication and coordination, and provide useful information for ongoing progress. The OB schedule was printed daily on paper, and any rescheduling had to be done manually. Staff manually logged time-related data such as operation start times, entry event into the recovery room (RR), and movement in and out of the compartment. Moreover, anamnestic data for patients was not considered. This critical patient information, which included medical history, allergies, and previous procedures, remained separate from the scheduling and treatment processes. Important patient characteristics that could affect treatment decisions and outcomes were frequently overlooked. This manual procedure was susceptible to errors, delays, and inefficiencies, resulting in suboptimal resource allocation and potential complications in patient care.

3.2.2 Requirements Analysis

Taking into account the various challenges outlined in the background and the concerns raised by staff, a comprehensive analysis of the functional requirements for the proposed system was performed. The key functional requirements are:

- **Automation of Scheduling Processes:** The system should fully automate the scheduling processes within the OB, replacing the manual paper-based methods to enhance efficiency and minimize human error.
- **Integration of Anamnestic Data:** The system must integrate patients' anamnestic data, including medical history and previous surgeries, ensuring that patient-specific factors are considered in scheduling and treatment decisions.
- **Real-time Tracking:** Implement real-time tracking capabilities to monitor patients' movements within the OB, improving coordination and minimizing de-

lays.

In this context, a survey of tracking technologies such as Ultra-Wideband (UWB) [100], Radio Frequency Identification (RFID) [101], and Bluetooth Low Energy (BLE) was carried out concerning tracking precision, power consumption, cost, and compatibility with infrastructure.

- UWB: it had highly valuable accuracy but was very power-consuming and extremely expensive, hence unsuitable for use in a resource-constrained environment.
- RFID was energy-efficient and cost-effective but lacked the real-time tracking accuracy required in dynamic settings such as the OB.
- BLE was the most suitable option because it provided a trade-off between accuracy, energy efficiency, and cost, with easy integration into existing systems.

This evaluation informed the decision to adopt BLE as the foundation for the tracking system, aligning with the OB's requirements for scalability and compatibility.

- **Compatibility with Legacy Hardware:** Ensure smooth integration with existing legacy hardware systems within the OB, maximizing their functionality and overall operational efficiency.
- **Data Accuracy and Security:** Guarantee the accuracy and security of all data collected and stored, including patient information, scheduling details, and operational metrics, to protect against unauthorized access and ensure reliability.
- **Historical Data Analysis:** Enable the analysis of historical data to identify trends, patterns, and areas for improvement within the operational block, thereby enhancing decision-making processes.
- **User-Friendly Interface:** Provide an intuitive and user-friendly interface tailored to the needs of various user groups, including administrative, medical, and support staff, to facilitate ease of use and efficiency.

- **Scalability and Flexibility:** Design the system to be scalable and adaptable, allowing for future developments and changes within the operational block, including the addition of new features and integration with other systems.

3.2.3 DT Architecture in the Operating Block

With all the above criteria presented in Section 3.2.2, A DT architecture was designed for the OB and seamlessly integrated into the existing hospital legacy system. The prototype supplies a certain number of key functionalities for optimization of OB management in the scope of:

- **Real-time Monitoring**: The DT allows for continuous tracking of patients and monitoring of surgical procedures inside the OB.
- **Predictive Analytics**: The DT will analyze current data and past trends to provide predictive insights that help identify complications that may arise from surgery, hence offering a proactive intervention for better patient outcomes.
- **Decision Support**: The DT allows for decisions supported by analytics that will help the medical teams make informed choices on patient care.
- **Optimization of Resources**: The DT works out and improves workflows and resources in real-time to minimize delays, thereby reducing operational costs while maximizing overall efficiency in the surgical block.

DTs enable the development of systems that provide OB management, surgical intervention planning, and control of critical resources. The paradigm of a DT allows capturing in real time all data from the real scenario, while all the historical information is maintained within an appropriate database. All this information enables the tracking and modeling of evolution regarding conditions of operating and RRs, enabling the performance of several scenarios to find the most optimal outcomes.

The DT in this project was realized by three main layers, all enablers of a fully functional integrated system:

- Physical Layer: It depicts the physical elements present in the operational

block. The components would be everything in physical form, from the patients to health professionals, medical equipment to rooms, surgical operation theatres, and RRs. Anything that may affect the workflow in surgery is part of this layer. Later, the discussion will cover the BLE IoT Tracking System, and it works primarily at this physical layer, continuously providing information on patients in terms of location and movement.

- Digital Layer: It models the physical world in real-time through the creation of a real-world, virtual model. Data continuously flows from the physical into the digital layer, allowing the DT to display in real-time the state of the OB with a high degree of fidelity and accuracy. This digital model is dynamic, not static; it changes along with continuous processes taking place in the real world. This digital layer keeps not only real-time updates but also the data that has built up over time, thereby allowing retrospective analysis and long-term planning.

- Communication Layer: It provides an easy flow between the physical and the digital layers. The communication layer is based on a secure, and efficient communication protocol such as MQTT or Message Queuing Telemetry Transport. These protocols ensure data from different IoT sensors, devices, and other hospital systems can move to the digital environment in real-time. This layer plays the most crucial role in keeping both the physical and the digital worlds aligned with each other; thus, this provides a high degree of efficiency to DTs for operation with minimum latency.

Further integrations of AI algorithms can be applied to reinforce the performance of the model and to plan effectively all the resources. This approach creates a virtual environment that mirrors the physical world, where the OB is represented as a DT containing all essential information. With this DT, it's possible to simulate real-life conditions, track staff, and patients, and optimize the management of ORs and RRs. Figure 3.1 illustrates our DT architecture, which integrates real-world data into the DT layer. This approach ultimately improves operational efficiency and enhances patient care.

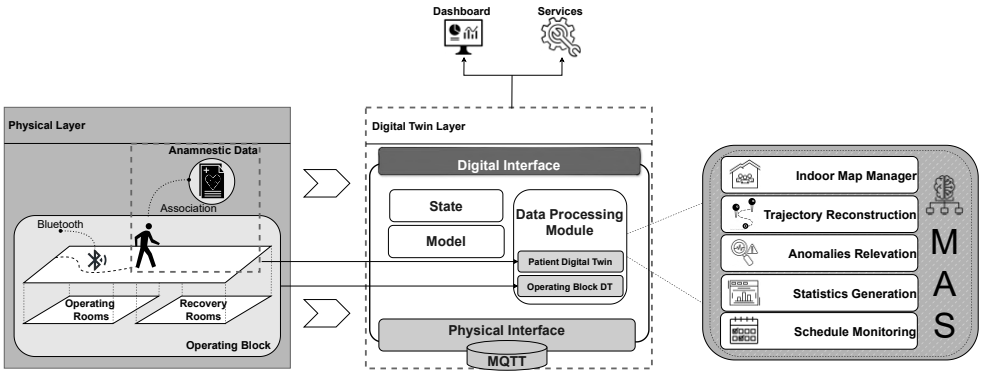

Figure 3.1: A graphical illustration of our DT architecture shows real-world data integration to improve operational efficiency and patient care in operating and RRs.

3.2.4 Physical Layer

It refers to the real-world components within the OB at the hospital. It involves all the physical or tangible entities: the patients, the medical staff, equipment, and physical spaces where operations take place, such as ORs and RRs. Central to the physical layer is the BLE IoT Tracking System, which provides real-time location and status of the patients and resources of the OB. To collect patient tracking data in the operating compartment, minimizing human error, such architecture was developed to perform indoor localization of patients in the operating compartment.

Several key elements of the physical layer can be identified, each playing a critical role in ensuring the reliable transmission and reception of data across communication networks:

- **Patients**: Patient modeling is an integral part of this layer, where individual patient profiles are created based on their medical history, surgical requirements, and real-time data. This modeling helps predict patient outcomes and optimize care pathways.

- **BLE Beacons**: Each patient is equipped with a BLE beacon embedded in a

wristband. The lightweight design of the BLE beacon ensures that it does not interfere with patient comfort or medical procedures. This beacon is a critical part of the IoT tracking system, transmitting signals that allow for continuous monitoring of the patient's location throughout the OB. These send signals in periodic intervals, which can easily be captured by the sensors in their proximity to track a patient's location across the OB.

- **Raspberry Pi Detectors**: These are at strategic positions within the OB. It acts like sensors that detect BLE signals. The sensor records entry event time, exit event time, and further movements in and out between OR and RR whenever any beacon is detected.
- **Central Server**: Collected data from the detectors will be pushed to a central server over an MQTT protocol. It then aggregates the data and stores it in a MongoDB database to give a full view of patient mobility and utilization of rooms.
- **Operating Rooms and Recovery Rooms**: The physical layer also includes the physical spaces where medical procedures and post-operative care take place. The ability to track when a patient enters or leaves these spaces provides critical data for resource management, such as determining when an OR is available for the next surgery or when a patient is ready to be transferred from the RR.
- **Medical Equipment**: While patient tracking is central to the BLE IoT system, the tracking of certain critical equipment within the OB is also possible. For example, specialized surgical instruments or monitoring devices can be equipped with their own BLE tags, allowing staff to quickly locate and allocate these resources as needed.

Patients who choose to participate in the study are invited to provide informed consent. If they agree, they receive a personal BLE tag upon entering the OB or hospital area. The tags are detected by detectors positioned within the area of interest. The detectors communicate with a private local area network (LAN) to manage the data flow and provide additional layers of security. Furthermore, the LAN has no Inter-

net access, making the network immune to external attacks. Access to the server, the sensor configuration module, and the association interface are password-protected, and all communication with the server is encrypted. A client-server architecture was implemented to facilitate communication between the sensor modules and the server in the system.

To respect patient privacy, each patient is assigned a beacon linked to a numerical, progressive, and anonymous ID. The beacon is worn as a wristband, representing the best compromise between comfort and signal efficiency.

As a sensor module, a Raspberry Pi 4 model B [102] device running the 64-bit Raspbian OS was used. The sensor module continuously scans for advertising-type BLE packets, verifying them against a list of pre-registered beacons provided by the central server. The use of a list allows the system to ignore unregistered beacons, saving bandwidth and processing resources. The sensors self-configure and can be restarted or powered down at any time. When the sensors receive a signal from a beacon, they add the received packet and the corresponding timestamp to a JSON file and forward the information to the central server via Transmission Control Protocol (TCP).

The patient's presence within a specific room in the OB is assessed by measuring the duration of their stay within the sensor's range. Each sensor has its own duration and an adjustable signal sensitivity threshold. This parameter can increase or decrease the sensor's range. If the beacon remains within the sensor's range for a certain period, it is marked as having entered the room. Figure 3.2 shows a graphical representation of a generic patient's pathway within the operating area. A generic patient accesses the operating area from point 1. At point 2, the patient is placed on another stretcher, and an operator assigns them a wristband and handles the relevant association; subsequently, the patient enters the assigned OR, for instance, at point 3. Once the surgical procedure is completed, the patient is transferred to the RR at point 5, where they are monitored post-surgery. The patient can then exit the operating area at point 6 or return to the OR for additional surgical procedures. Furthermore, during these activities, the patient may remain inactive at an unspecified point in the corridor, represented by point 4.

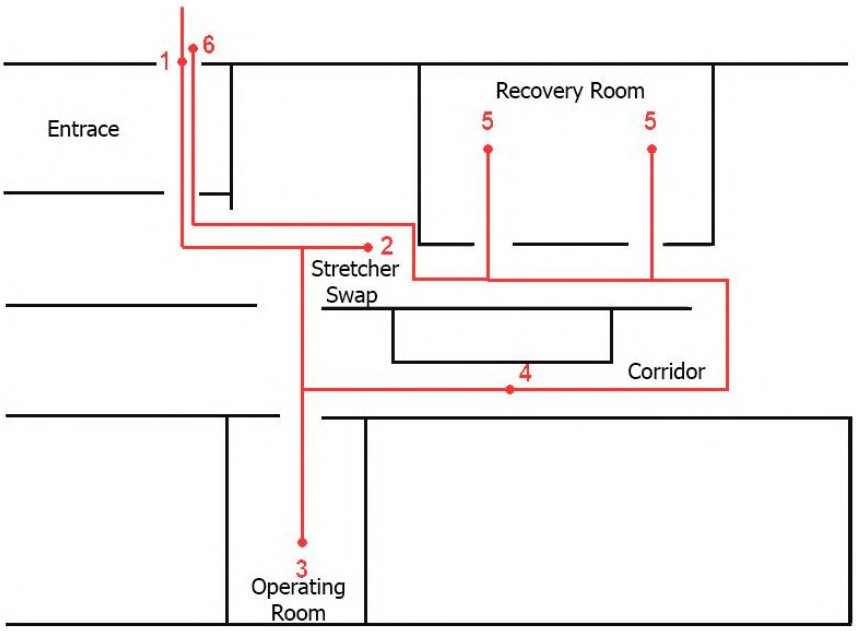

Figure 3.2: Diagram of the typical patient journey in the OB

The central server indexes and collects data from the sensor modules. Specifically, it has the following tasks:

- Store records from each detector in a MongoDB (MongoDB Inc) database.
- Coordinate the distributed solution and message exchange using a publish-subscribe mechanism based on the Message Queuing Telemetry Transport protocol.
- Expose a web server that serves as a single interface between the software architecture and the hospital medical staff.

The central server hosts the MQTT broker based on Eclipse Mosquitto (Eclipse Foundation Inc). When it receives packets from the sensors, it determines whether the beacon is located in the room where the sensor is positioned. The beacon server

implementation used a Python-based service and leveraged MongoDB to store various events. The central server also has the task of sending information to the edge modules regarding the list of pre-registered beacons, the sensor's identity, and a time synchronization message.

Signal Loss Tolerance-Based Mechanisms To evaluate the system's robustness, an investigation of potential lost beacon signal failures is performed. Among 1,300 patients, 10.5% have beacon signal loss due to external interference, battery depletion, or temporary obstructions, such as medical equipment masking signals. The constructed system's interpolation function recovers 10.5% of the lost data. To accurately reconstruct patient flow, the remaining 89.5% of signal losses resulted in a mean variation of ± 2.1 minutes, well within the allowable error tolerance of ± 5 minutes. These data demonstrate that, despite occasional signal loss, the architecture's design is reliable in detecting patient movements with low influence on operational efficiency.

A tolerance-based stationary time method is used to make the system more resilient when a beacon signal gets lost. This method is designed to mitigate the impact of short-term signal losses and provide accurate patient tracking even if a beacon briefly stopped providing data. For each monitored room, a stationary time limit is set. The cutoff time for ORs was set to 5 minutes. This means that if a beacon signal is lost, the system will still consider the patient to remain in the OR for 5 minutes after the last signal is received. The recovery room (RR) threshold is chosen at 10 minutes to account for possible longer delays caused by medical equipment positioning, patient movements, or other interference sources.

The system base logic ensure to capture the first entry event for each tag (or user). For leave events, however, the system uses the same stationing-time criterion in reverse: if beacon signals are not detected within a predefined threshold (either 5 or 10 minutes, depending on the room), the patient is identified as having left the room. This strategy allows it to miss a considerable amount of beacon signals while maintaining the integrity of the data acquired. Even when communication is suspended for an extended period of time, the system continues to detect real-time patient positions

with great accuracy. Using this tolerance strategy, false exit events are eliminated, preventing overestimation or underestimation of room occupancy duration.

Furthermore, the technique works best in operating rooms, where transient signal interruptions can be generated by nearby surgical equipment, anesthetic equipment, or other clinical devices interfering with beacon transmissions. Similarly, for the RR, where patients are stationary for extended periods of time, the higher 10-minute threshold prevents premature exit event tagging, allowing for more accurate and realistic reporting of patient mobility.

Despite up to 10.5% signal loss, the system shows over 90% accuracy in patient transfer tracking. This demonstrates that our digital twin technology, when combined with Bluetooth Low Energy tracking and tolerance-based techniques, is resilient and reliable in dealing with typical issues like intermittent beacon failure

Integrating Anamnestic Data In the preoperative phase, several key patient variables are critical for thorough assessment and planning. Baseline demographic details like age, weight, and height can be obtained from historical records. This information is vital for calculating anesthesia dosages and assessing patient risks. Electronic health records (EHRs) also provide insights into comorbidities, such as cardiovascular issues, respiratory problems, and diabetes. Understanding these factors helps determine the best type of anesthesia and the appropriate perioperative management strategies.

During surgery, real-time patient monitoring is crucial. It's important to integrate anesthetic data with movement tracking to ensure patient safety and optimize surgical outcomes. By continuously observing vital signs like heart rate, respiratory rate, and oxygen saturation alongside anesthesia-specific metrics, anesthesiologists can swiftly identify and respond to any deviations from the desired physiological state. Improving situational awareness by incorporating historical data on surgical procedures and patient positioning can help healthcare teams foresee and avert potential complications during surgery.

Post-surgery, anesthetic data can be combined with tracking of patient movements to oversee and manage recovery. Key parameters to monitor include total re-

covery time, length of hospital stay after surgery, and any complications that may arise. Analyzing this information alongside previous records of Intensive Care Unit (ICU) admissions, planned ICU stays, and hospital readmissions allows healthcare providers to identify trends and implement targeted interventions to enhance post-operative care and reduce costs. This strategy can optimize recovery pathways and improve patient outcomes.

Integrating anesthesia data with historical patient movement data in the OR offers a significant opportunity to enhance management and care delivery. By utilizing advanced analytics and ML, predictive models can be developed to forecast patient outcomes, optimize resource use, and streamline perioperative workflows. Furthermore, real-time data integration supports proactive decision-making, facilitating early interventions and minimizing the risk of adverse events. Finally, healthcare facilities can improve operational efficiency, boost patient safety, and enhance surgical outcomes through effective data integration and analytics.

3.2.5 Digital Twin Layer

The DT layer can do something much more detailed than just collecting data. It acts as one source of truth about our operational environment, linking the physical to the virtual and smoothly incorporating information from many sources. It queries a central server regularly to stay updated with the real world. Predefined rules are implemented to quickly identify critical issues, automatically alerting relevant stakeholders who require timely action.

With this capability, the DT gains spatial awareness, whereby it can keep track of the ongoing surgical procedures and room availability status in a room in near real-time. This spatial intelligence is enhanced through forecasts of demand that provide valuable insights for better scheduling and resource allocation. These, in turn, enhance efficiency in operations.

The proposed model, as shown in Figure 3.1, is made of four interrelated components, the DT layer, to support efforts towards better patient care while improving operational efficiency. First, the physical interface couples the DT system with the physical environment, allowing for real-time monitoring and control of patient health

and movement data, as well as equipment performance. Second, the state component retains the current status and configuration of the system, including significant patient medical histories and operational data.

The model component analyzes, in turn, the behavior of the physical system with computational models. This helps us predict patient outcomes, smoothen workflows in the OR, and recognize potential issues before they grow. The data processing module completes this circle by providing aggregation and analysis to continuously improve both operational efficiency and patient outcomes. Even more is achieved with this module by using a multi-agent system; thus, it provides high-scale data gathering, processing, and analysis to enable better decision-making.

Data Processing Module A pivotal part of the DT layer is the data processing module. Such module forms the backbone of the system's ability to cope with the management and analysis of data within the framework of the OB. The module is designed with a distributed architecture, where multiple specialized agents perform different tasks. This is achieved by splitting the work amongst the agents so that the system operates more effectively, exploiting its resources optimally and adapting to real-time needs with minimal delay.

Each of the agents plays a critical role in ensuring the smooth running of the system, including data collection, processing, analysis, and triggering necessary actions. The agents and their responsibilities are described below:

- **Data Collection Agent:**
  - Each Raspberry Pi device within the OB is equipped with a data collection agent. It collects raw data from the BLE beacons worn by patients. This agent continuously monitors and fetches data from BLE signals, performs preliminary filtering and organization, and then sends the data to a central server.
- **Data Processing Agent:**
  - This agent takes over once the data has been collected. It cleans, formats, and prepares the raw data for further analysis. The agent also performs

basic analysis, such as noise filtering, thereby reducing the load on the central system and speeding up the entire data processing pipeline.

- **Location Tracking Agent:**
  - This agent determines the real-time location of patients within the OB. By processing data from multiple Raspberry Pi devices, it continuously calculates the precise location of each patient. This information is crucial for tracking patient movement between ORs, RRs, and other locations in the OB.
- **Alerting and Notification Agent:**
  - This agent monitors patient movements and triggers alerts under certain conditions. For example, if a patient remains in a RR for too long, the system will promptly notify medical staff. This helps ensure timely interventions and enhances patient safety.
- **Communication Agent:**
  - The communication agent manages all data exchanges between Raspberry Pi devices and the central server. It uses secure protocols like MQTT to ensure efficient and reliable data transmission across the system. This agent plays a key role in maintaining synchronization and preventing disruptions.
- **User Interface Agent:**
  - The user interface agent manages the dashboard that healthcare staff use to track patient movements. It provides an intuitive interface, displaying the real-time location of patients and alerts, making it easier for healthcare professionals to monitor the situation in the OB.
- **Analytics Agent:**
  - The analytics agent is responsible for performing deeper data analysis, generating reports, and identifying patterns in patient flow and room uti-

lization. By analyzing historical data, this agent helps optimize scheduling and predict trends, improving the overall efficiency of the OB.

- **Security Agent:**
  - The security agent ensures system security by managing encryption, access control, and monitoring for suspicious activity. This agent helps protect patient data and the overall system from security breaches.

The distributed nature of these agents is crucial to the system's modern design. By distributing tasks across several specialized agents, the system can efficiently handle high data volumes, adapt to real-time changes, and maintain resilience in case of failures. While such an architecture requires careful coordination and robust security measures, it provides the necessary flexibility and scalability to meet the demands of an active OB, ultimately ensuring smooth operations and improving patient care.

3.2.6 Dashboard

The proposed architecture encompasses a user-friendly web-based GUI acting as a control panel to our architectural framework. It is implemented on top of the DT layer, where it makes difficult data easy for medical staff to understand. A strong security mechanism is set in place at the level of the GUI to protect the information for a particular patient. It also provides visualization tools for historical tracking data, enabling the observation of trends and changes over time. It also provides easy integration of patient medical history, improving the experience of practitioners using the system.

This innovative web application can only be accessible after stringent authentication. This is a major step to address accountability concerns, ensuring that only authorized medical personnel will be able to modify or register new information about surgeries and patient records. By doing this, a focus on advanced healthcare management technology is essential for improving outcomes and processes in the medical environment.

The client interface, hosted by the central server, offers several functionalities.

The first is registration. Patient registration is performed by a member of the medical staff when the patient enters the OB for a surgical procedure. Registration occurs using a tablet device that communicates with the web application and provides the necessary information, associating the BLE bangle with the patient. The system also offers a graphical interface that monitors patients' movements in real-time, as shown in Figure 3.3.

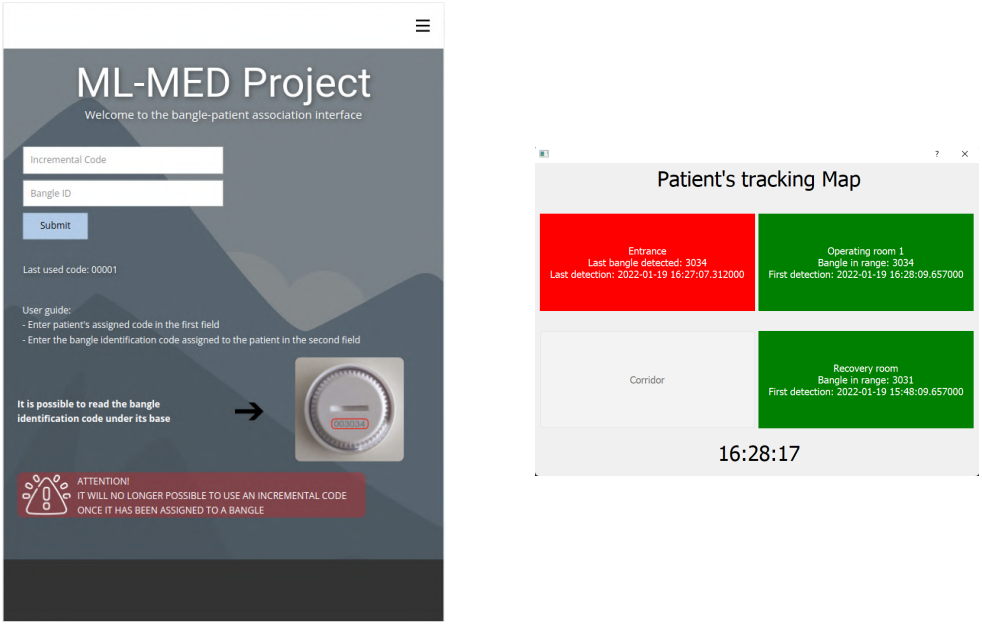

Figure 3.3: Interface used by hospital staff to handle all tasks required for proper patient registration and follow-up

On the other hand, Figures 3.4 and 3.5 give an illustration of our management dashboard and consequently bring forth the underlying power. Figure 3.4 is the reconstruction of patient histories using anchor points of variable positions in the OB. Every anchor point records key time instants, which include the time of the patient's first detection and the time of his/her departure. This allows the tracking of patients' routes while in the OB. To represent a patient's state in real-time, a graphical flag was

Figure 3.4: Tracking Patient Paths: A visual display of the patient's journey through the OB's layout.

employed beside each patient's unique ID flag, which is green if the patient is active and turns red if they are inactive. This visualization gives supplementary operational insight and supports both room management and patient management.

In Figure 3.5, a set of statistical information is presented, obtained through the analysis of historical patient data. The system contains a wide variety of information and metrics, such as the total count of surgeries performed in day-to-day and month-to-month analysis. Moreover, it gives the ability to comprehend how trends in room occupation work by showing the average amount of time taken in days, weeks, and

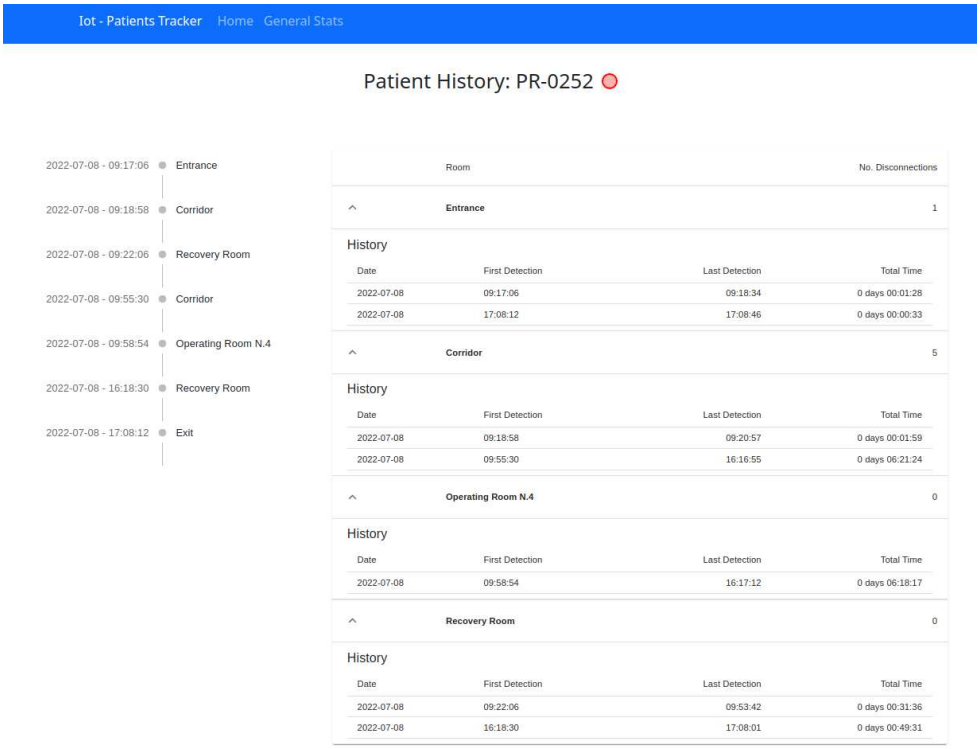

Figure 3.5: A web app view featuring insightful graphs for analyzing the utilization of OB spaces.

months. Detailed statistical analysis of this type at this level makes medical staff more knowledgeable in making decisions, thus helping in effective resource allocation and overall better operational efficiency.

3.2.7 Results

The implementation of the DT architecture at *Ospedale Maggiore di Parma* has been a significant milestone in optimizing OR management. Our method covers a variety of components that form a coherent framework for scheduling and, therefore, carry out valid data acquisition, modeling, and storage, allowing valid decision-making.

After a short testing and tuning period, data from 1,300 patients were collected and times recorded in the EHRs were compared to those obtained from the BLE architecture. Three different cases of time spent are considered: totally inside the OB, in the OR, and in the RR.

It allows us to track patient mobility and operational efficiency because the DT architecture supplies a virtual representation of the physical environment on time. The data from the BLE system is considered "ground truth" to effectively review the accuracy of the records maintained in the EHR.

Table 3.1 reports the *root mean squared error (RMSE)*, the *mean absolute percentage error (MAPE)*, and the *standard deviation (STD)* between the data from the BLE architecture and the EHR (times are expressed in minutes). Additionally, Figure 3.14 illustrates the differences in recorded times (in minutes) between the BLE data and the EHR for each patient, providing a visual representation of the discrepancies.

The preliminary results indicate that our architecture significantly reduces the errors associated with manually acquired EHR records, which are often subject to human error and approximations. The analysis reveals that the time differences expressed in MAPE range from 11.48% in the OB to 16.09% in the OR and even 39.79% in the RR. Notably, the EHR data underestimate the occupation of the OB up to 59.66% of the time and overestimate it by as much as 23.53%. Additionally, the Mean Bias Error (MBE) metrics show that the EHR system overestimates BLE data in the OB by -4.46 minutes, while in the OR, it underestimates by 24.62 minutes, and in the RR, it overestimates by -13.26 minutes. These findings highlight the limitations of traditional record-keeping methods and underscore the advantages of implementing a DT framework enhanced by BLE tracking.

In this context, the MAPE metric reflects the magnitude of errors, which is critical in the quantification of the variability of the times of record. The percentage is higher; the larger the deviation, observed in RR, hence the area where improvements in precision are most needed. On the other hand, MBE provides information on the systematic biases of the data. For example, the negative MBE in the OB was -4.46 minutes, which means that the EHR system has consistently overestimated BLE data, leading to underutilization of resources within this setting. In contrast, the positive

MBE in the OR was 24.62 minutes, reflecting consistent underestimation that may lead to resource shortages and scheduling conflicts.

Data were considered with an error margin of ±5 minutes in alignment with the BLE detection capabilities, thus validating the measurements and further confirming the robustness of the architecture.

BLE - EHR	RMSE	MAPE	STD	MBE
Total occupation time - OB (in minutes)	67.672	11.48 %	65.918	-4.46
OR occupation time (in minutes)	30.515	16.09 %	29.405	24.62
RR occupation time (in minutes)	31.248	39.79 %	31.212	-13.26

Table 3.1: Comparison in terms of root mean squared error (RMSE), mean absolute percentage error (MAPE), standard deviation (STD), and mean bias error (MBE) between times recorded by the BLE architecture and those in the EHR recorded by the medical staff (times are expressed in minutes).

Cluster analysis

To further investigate the differences between BLE and EHR data, time differences in the OB, OR, and RR categories were analyzed using cluster analysis. K-means segmentation divided the data into three clear clusters.

The Elbow Method [103] was exploited to determine the optimal number of clusters. In this analysis, three was determined to be the best number of clusters that would yield meaningful groupings without overfitting or underfitting the data.

To better understand the clustering results, data were visualized using two complementary methods: Pair plot and 3D cluster visualization. Figure 3.6 shows a pair plot that illustrates the relationships between time differences in the OB, OR, and RR categories. On the other hand, Figure 3.7 provides a 3D scatter plot of the clusters, illustrating their spatial distribution across the time difference dimension.

The pair plot and 3D cluster visualization give complementary insight into the clustering results. Indeed, both the visualizations reveal Cluster 0 to closely linked with low discrepancies, Cluster 1 is moderately dispersed with average discrepancies,

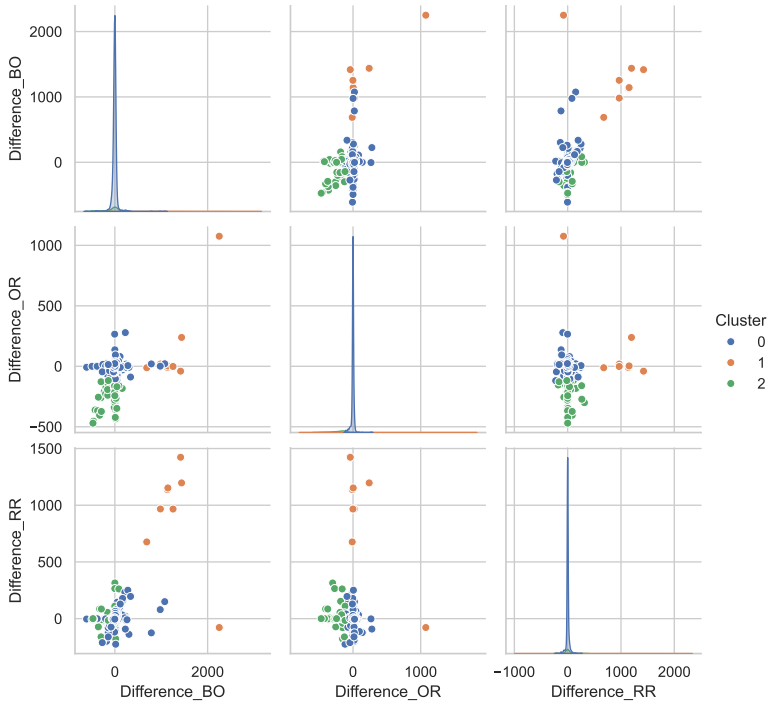

Figure 3.6: Pair Plot of Clusters showing time differences (BLE - EHR) for OB, OR, and RR categories. Clusters are color-coded to reflect distinct groupings.

and Cluster 2 is well separated with high discrepancies, especially in the RR category. While the pair plot shows the relationship among specific features, which are the time differences of OB, OR, and RR, the 3D scatter adds depth to that insight into the spatial distribution of the clusters and how well they separate from each other.

The pair plot shows the feature correlations and gives insight into how the clusters are positioned in relation to one another. On the other hand, the 3D visualization adds another dimension of spatial relation between the clusters and reaffirms the distinct separations of these clusters. It gives more clarity about the perception of dispersion in each cluster and the general placement of each within the time-difference space.

Cluster 0, which has minimal discrepancies, acts like a benchmark for an ideal

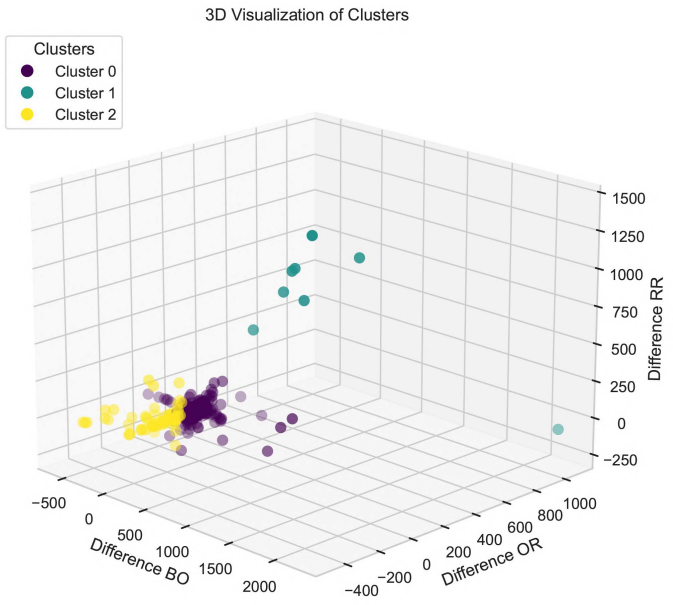

Figure 3.7: 3D Visualization of Clusters across time differences (BLE - EHR) for OB, OR, and RR categories. Cluster separation highlights distinct patterns of discrepancies.

BLE-EHR integration where the systems set and work together. This cluster can act as a model for best practices, showing where the integration performs optimally. Cluster 1, which had moderate discrepancies, pointed to areas that needed improvements in refining processes, providing additional training to staff, or fine-tuning workflows. Lastly, Cluster 2 with high discrepancies indicates critical outliers. The distinct separation of this cluster would indicate that it contains cases with serious errors, system problems, or other unusual events, all of which require further investigation and resolution.

These visualizations provide a strong comprehension of the clustering results through pairwise and spatially enhanced views. They confirm the trends and set the context for interpreting the data discrepancies, enabling targeted improvements in system integration, process optimization, and overall hospital operations. The in-

sights from both these visualizations are key to hospitals for improving operational efficiency, reducing discrepancies, and ultimately improving patient care.

3.3 Final Remarks

In conclusion, the introduction of DT architecture at "Ospedale Maggiore di Parma" performed very well and was originally for OR workflow management. The feasibility of a BLE IoT tracking system in developing a virtual environment synchronized with reality has been demonstrated; this not only increases the accuracy of patient tracking but also reduces the chances of errors in the Electronic Health Records recorded manually.

Our findings indicate that the DT framework is more reliable in capturing the dynamics of the OB; hence, it allows for more complex scheduling and resource allocation. The large discrepancies identified between the BLE data and the EHR emphasize limitations in traditional manual record keeping and point toward possible AI-driven insight that might result in higher-quality care and better operational efficiency.

In the future, the study will refine the data analysis module for efficient support in surgery scheduling and patient care. The aim is to leverage the data gathered through the system to develop advanced AI-based functionalities within the DT framework. In particular, ML algorithms are intended to be adopted to predict surgical durations., starting from pre-operative patient data, types of surgical procedures, and optimal team compositions. Furthermore, diverse patient data will be explored to conduct risk assessments before, during, and after surgeries. This continuous learning curve ensures that new patient data keeps getting added to our model at regular intervals, with increased refinement in scheduling processes and optimization of resources.

With these activities, the goal is to demonstrate that the DT paradigm, supported by IoT tracking systems, is effective, feasible, and necessary to improve overall OR management and healthcare.

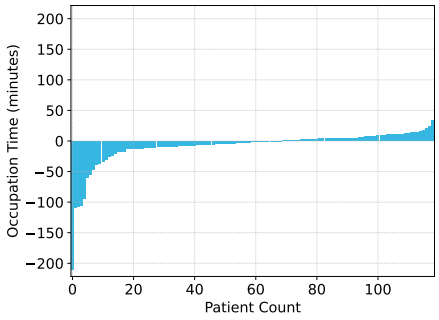

Figure 3.8: OR time differences

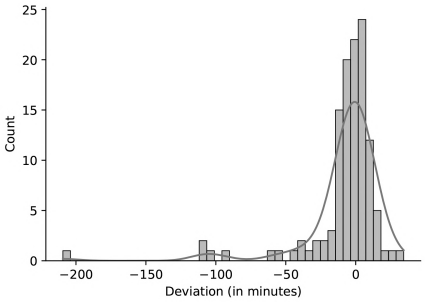

Figure 3.9: OR error distribution

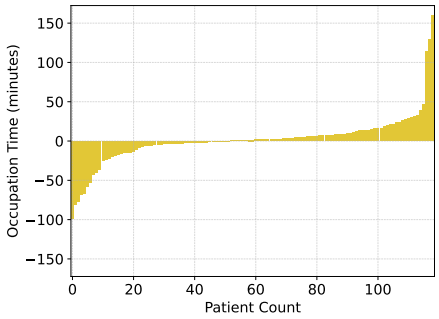

Figure 3.10: RR time differences

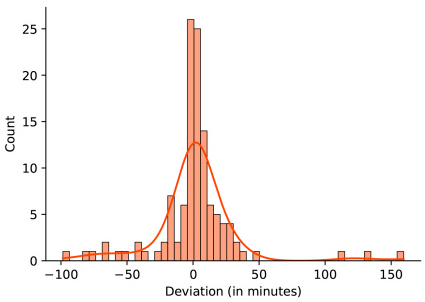

Figure 3.11: RR error distribution

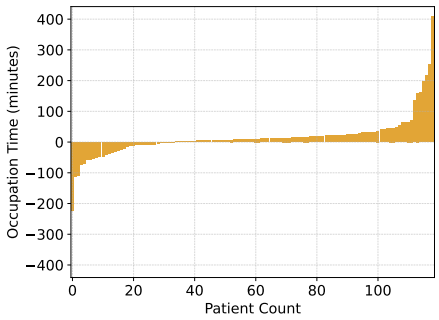

Figure 3.12: OB time differences

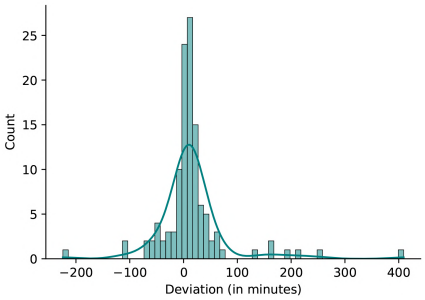

Figure 3.13: OB error distribution

Figure 3.14: Representation of differences between BLE and EHR data in OR, RR, and OB (in minutes) for each patient, along with error distribution. Positive values in (a), (c), and (e) indicate EHR underestimation, while negative values indicate overestimation

Chapter 4

Conclusions

In conclusion, this thesis has explored the macro-theme of artificial intelligence in healthcare, focusing on various aspects ranging from epidemic simulation to cyber-physical systems in medicine. These topics are very much interlinked, whereby epidemic simulation provides essential insights into the spread of disease and resource distribution. At the same time, CPSs enhance real-time data collection and response strategies within healthcare environments. These approaches collectively enable the retrieval and optimization of information that can be used to develop better decision-making and operational efficiency in healthcare contexts. This research covers significant challenges in healthcare systems through developing and applying advanced computational methodologies, moving toward an integrated and responsive framework for health services.

The first part of the work focused on Agent-Based Modeling and Simulation (ABMS) for epidemic scenarios, offering detailed insights into how multi-agent systems can simulate complex social interactions and disease dynamics. The proposed ActoDemic framework, applied specifically to the COVID-19 pandemic, demonstrated the ability to model individual behaviors and interactions, yielding fine-grained simulations that are crucial for predicting disease spread and evaluating containment strategies. The research highlighted the importance of modular and adaptable approaches, with further refinement in modeling social behaviors and transmission dy-

namics, especially for specific regions like Lombardy and Emilia-Romagna. This work offers a promising foundation for future epidemic modeling, with potential for integrating more refined behavioral and mobility models to improve predictions and policy recommendations.

Future research should focus on expanding these models with more specific regional data, such as mobility patterns in Lombardy and Emilia-Romagna. It should also integrate additional layers of behavioral and environmental factors that could enhance prediction accuracy and policy recommendations. Future studies may wish to address how ABMS might be further enhanced by including real-time data from ongoing epidemics to enable responsive simulations.

The second part of the thesis shifted toward CPSs in healthcare, particularly focusing on the Digital Twin concept applied to operating rooms. Through detailed analysis and the use case of Ospedale Maggiore di Parma, the thesis presented a robust framework that integrates real-time data through IoT devices. The Digital Twin architecture enhances operational efficiency and patient care, offering a pathway to more intelligent, data-driven healthcare environments. The use of BLE and other IoT-based systems for real-time tracking and management of hospital workflows is a notable advancement that aligns with the broader trend of smart healthcare systems.

Future research should expand the use of Digital Twins beyond operating rooms to other life-supporting areas within a hospital, such as emergency departments, intensive care units, and diagnostic labs. Such extension will further enhance healthcare operations, patient monitoring, resource allocation, and allow personalized treatment plans. Finally, as data is accrued, further work could be directed toward incorporating advanced AI models within the architecture of Digital Twins to take real-time decision-making and operational insight to the next level.

Despite the contributions of this research, several limitations inherent to the proposed methodologies must be acknowledged. First, concerning the ABMS, there is an inherent limitation in most cases concerning the availability of real-world data. In many cases, the data on social interaction and behavioral dynamics are incomplete or too simplified; therefore, any model developed may not capture the full complexity of human behaviors. For example, static representations of social connectivity

cannot capture the dynamic changes that occur during public health interventions, such as the shifting levels of compliance with lockdown measures. Such simplifications might introduce biases in the predictions of epidemic spread and, thus, probably make the policy recommendations derived from such models less robust. Moreover, the computational intensiveness of high-fidelity simulations requires trade-offs between model complexity and execution time, thus limiting the granularity of insights provided.

Cyber-Physical Systems (CPS) also face unique challenges that impact their practical deployment and reliability in healthcare settings. Real-time data integration through IoT devices has potential risks: inaccurate sensors and network latency. These sensors may eventually give incorrect readings, such as underreporting vital signs of a patient or environmental conditions, which could lead to a false alarm or the failure to notify critical events. Additionally, segmentation and loss in communications among sensors can also degrade decision making such as over operation theaters or critical care units-their effectiveness essentially binds to real-time responses. All these potential inefficiencies require solid mechanisms in the CPS architecture for robust validation, calibration, and fault tolerance if these need to obtain reliable applications in healthcare in real environments.

Future work to address these limitations will require interdisciplinary efforts to improve data collection methodologies, refine behavioral models for ABMS, and enhance IoT infrastructure to reduce errors and ensure seamless data flow. Overcoming these challenges will improve the robustness and reliability of these systems, leading to more accurate and actionable insights in epidemic modeling and healthcare operations.

Looking forward, this thesis opens several avenues for future research. For ABMS, further exploration into integrating various datasets and refining social interaction models can enhance the accuracy of simulations in epidemic contexts. For CPS in healthcare, expanding the Digital Twin concept beyond operating rooms to other hospital functions, coupled with continuous advancements in IoT and AI integration, promises to transform healthcare operations. In the future, Digital Twins will integrate AI capabilities once a certain volume of data is available, enabling more

sophisticated analysis and decision-making processes that enhance operational efficiency and patient care. Both lines of research emphasize the need for interdisciplinary collaboration, combining expertise from AI, medicine, and engineering to achieve the full potential of these technologies in improving both public health and healthcare systems. Moreover, the creation of synthetic data is also a significant area for further study, as it might be essential for both ABMS and CPS applications. To improve model training, validation, and robustness, synthetic data generation can be a good substitute for real-world healthcare data, which is difficult to get due to privacy and availability issues. While CPS designs, especially Digital Twins, can employ synthetic datasets to improve decision-making and develop predictive algorithms, ABMS techniques can be used to generate realistic synthetic populations and model different scenarios. Without sacrificing patient privacy, using synthetic data techniques will not only increase the simulations' dependability but also make it easier for AI-driven healthcare solutions to be adopted more widely.

This thesis has laid the groundwork for ongoing advancements in epidemic modeling and healthcare systems, with the potential to significantly impact predictions, management, and responses to future challenges in both domains.

Chapter 5

Publications related to this thesis

Further information related to the works presented in this thesis, can be found in the following publications:

- Pellegrino, M., Lombardo, G., Mordonini, M., Tomaiuolo, M., Cagnoni, S., & Poggi, A. (2021). *ActoDemic: A Distributed Framework for Fine-Grained Spreading Modeling and Simulation in Large Scale Scenarios*. In WOA (pp. 194–209).

- Pellegrino, M., Lombardo, G., Cagnoni, S., & Poggi, A. (2022). *High-performance computing and ABMS for high-resolution COVID-19 spreading simulation*. Future Internet, 14(3), 83.

- Lombardo, G., Pellegrino, M., Tomaiuolo, M., Cagnoni, S., Mordonini, M., Giacobini, M., & Poggi, A. (2022). *Fine-grained agent-based modeling to predict COVID-19 spreading and effect of policies in large-scale scenarios*. IEEE Journal of Biomedical and Health Informatics, 26(5), 2052–2062.

- Pellegrino, M., Lombardo, G., & Poggi A. (2024). *Towards Digital Twins in Healthcare: Optimizing Operating Room and Recovery Room Dynamics*. In KES (International Conference on Knowledge-Based and Intelligent Information & Engineering Systems) 2024.

- Bottani, E., Bellini, V., Mordonini, M., Pellegrino, M., Lombardo, G., Franchi, B., Craca, M., Bignami, E., & others (2023). *Internet of Things and New Technologies for Tracking Perioperative Patients With an Innovative Model for Operating Room Scheduling: Protocol for a Development and Feasibility Study*. JMIR Research Protocols, 12(1), e45477.

- Pellegrino, M., Lombardo, G., Mordonini, M., Cagnoni, S., Bottani, E., Bellini, V., Bignami, E., & Poggi, A. (2023). *A System for Tracking Patients in the Operating Room—A Pilot Study*. In WOA (pp. 181–190).

- Bottani, E., Franchi, B., Monferdini, L., Bigliardi, B., Mordonini, M., Pellegrino, M., Lombardo, G., Collini, L., Bellini, V., & Bignami, E. (2022). *Opportunities for Simulation-Based Optimization in a Hospital Environment: Results and Perspectives From a Review of the Literature*. International Journal of Practical Healthcare Innovation and Management Techniques (IJPHIMT), 9(2), 1–21.

Bibliography

[1] Ross J Burton, Mahableshwar Albur, Matthias Eberl, and Simone M Cuff. Using artificial intelligence to reduce diagnostic workload without compromising detection of urinary tract infections. *BMC medical informatics and decision making*, 19:1–11, 2019.

[2] Xin Yang, Yifei Wang, Ryan Byrne, Gisbert Schneider, and Shengyong Yang. Concepts of artificial intelligence for computer-assisted drug discovery. *Chemical reviews*, 119(18):10520–10594, 2019.

[3] Bertalan Meskó, Zsófia Drobni, Éva Bényei, Bence Gergely, and Zsuzsanna Győrffy. Digital health is a cultural transformation of traditional healthcare. *Mhealth*, 3, 2017.

[4] Sobia Hamid. The opportunities and risks of artificial intelligence in medicine and healthcare. 2016.

[5] Jay W Forrester. Dynamic models of economic systems and industrial organizations. *System Dynamics Review: The Journal of the System Dynamics Society*, 19(4):329–345, 2003.

[6] William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

[7] Charles M Macal and Michael J North. Tutorial on agent-based modeling and simulation. In *Proceedings of the Winter Simulation Conference, 2005.*, pages 14–pp. IEEE, 2005.

[8] Muaz Niazi and Amir Hussain. Agent-based computing from multi-agent systems to agent-based models: a visual survey. *Scientometrics*, 89(2):479–499, 2011.

[9] Stefania Bandini, Sara Manzoni, and Giuseppe Vizzari. Agent based modeling and simulation: an informatics perspective. *Journal of Artificial Societies and Social Simulation*, 12(4):4, 2009.

[10] Franziska Klügl and Ana LC Bazzan. Agent-based modeling and simulation. *Ai Magazine*, 33(3):29–29, 2012.

[11] Stephen Wolfram. Cellular automata as models of complexity. *Nature*, 311(5985):419–424, 1984.

[12] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[13] H Randy Gimblett. *Integrating geographic information systems and agent-based modeling techniques for simulating social and ecological processes*. Oxford University Press, 2002.

[14] M Parker. Ascape: an agent-based modeling framework in java. In *Workshop onAgent Simulation: Applications, Models, and Tools*, page 149, 1999.

[15] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527, 2005.

[16] Michael J North, Nicholson T Collier, and Jerry R Vos. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 16(1):1–25, 2006.

[17] Andrei Borshchev. Multi-method modelling: Anylogic. *Discrete-event simulation and system dynamics for management decision making*, pages 248–279, 2014.

[18] U Wilensky. *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with Netlogo*. The MIT Press, 2015.

[19] Dmytro Chumachenko, Ievgen Meniailov, Kseniia Bazilevych, and Tetyana Chumachenko. On intelligent decision making in multiagent systems in conditions of uncertainty. In *2019 XIth International Scientific and Practical Conference on Electronics and Information Technologies (ELIT)*, pages 150–153. IEEE, 2019.

[20] Mikola Lysenko and Roshan M D'Souza. A framework for megascale agent based model simulations on graphics processing units. *Journal of Artificial Societies and Social Simulation*, 11(4):10, 2008.

[21] Tapas K Som and Robert G Sargent. Model structure and load balancing in optimistic parallel discrete event simulation. In *Proceedings Fourteenth Workshop on Parallel and Distributed Simulation*, pages 147–154. IEEE, 2000.

[22] Joshua M. Epstein and Robert L. Axtell. *Growing Artificial Societies*, volume 1. Mit Pr, 1999.

[23] Fabian Lorig, Emil Johansson, and Paul Davidsson. Agent-based social simulation of the covid-19 pandemic: A systematic review. *Journal of Artificial Societies and Social Simulation*, 24(3), 2021. URL: `http://dx.doi.org/10.18564/jasss.4601`, `doi:10.18564/jasss.4601`.

[24] H Van Dyke Parunak, Robert Savit, and Rick L Riolo. Agent-based modeling vs. equation-based modeling: A case study and users' guide. In *International workshop on multi-agent systems and agent-based simulation*, pages 10–25. Springer, 1998.

[25] Hazhir Rahmandad and John Sterman. Heterogeneity and network structure in the dynamics of diffusion: Comparing agent-based and differential equation models. *Management Science*, 54(5):998–1014, May 2008. URL: http://dx.doi.org/10.1287/mnsc.1070.0787, doi:10.1287/mnsc.1070.0787.

[26] Marco Ajelli, Bruno Gonçalves, Duygu Balcan, Vittoria Colizza, Hao Hu, José J Ramasco, Stefano Merler, and Alessandro Vespignani. Comparing large-scale computational approaches to epidemic modeling: Agent-based versus structured metapopulation models. *BMC Infectious Diseases*, 10(1), June 2010. URL: http://dx.doi.org/10.1186/1471-2334-10-190, doi:10.1186/1471-2334-10-190.

[27] Petrônio C.L. Silva, Paulo V.C. Batista, Hélder S. Lima, Marcos A. Alves, Frederico G. Guimarães, and Rodrigo C.P. Silva. Covid-abs: An agent-based model of covid-19 epidemic to simulate health and economic effects of social distancing interventions. *Chaos, Solitons amp; Fractals*, 139:110088, October 2020. URL: http://dx.doi.org/10.1016/j.chaos.2020.110088, doi:10.1016/j.chaos.2020.110088.

[28] Robert Hinch, William J. M. Probert, Anel Nurtay, Michelle Kendall, Chris Wymant, Matthew Hall, Katrina Lythgoe, Ana Bulas Cruz, Lele Zhao, Andrea Stewart, Luca Ferretti, Daniel Montero, James Warren, Nicole Mather, Matthew Abueg, Neo Wu, Olivier Legat, Katie Bentley, Thomas Mead, Kelvin Van-Vuuren, Dylan Feldner-Busztin, Tommaso Ristori, Anthony Finkelstein, David G. Bonsall, Lucie Abeler-Dörner, and Christophe Fraser. Openabm-covid19—an agent-based model for non-pharmaceutical interventions against covid-19 including contact tracing. *PLOS Computational Biology*, 17(7):e1009146, July 2021. URL: http://dx.doi.org/10.1371/journal.pcbi.1009146, doi:10.1371/journal.pcbi.1009146.

[29] Janos Gabler, Tobias Raabe, and Klara Röhrl. People meet people: A microlevel approach to predicting the effect of policies on the spread of covid-19. 2020.

[30] Christopher Wolfram. An agent-based model of covid-19. *Complex Systems*, 29(1):87–105, April 2020. URL: http://dx.doi.org/10.25088/ComplexSystems.29.1.87, doi:10.25088/complexsystems.29.1.87.

[31] Md. Salman Shamil, Farhanaz Farheen, Nabil Ibtehaz, Irtesam Mahmud Khan, and M. Sohel Rahman. An agent-based modeling of covid-19: Validation, analysis, and recommendations. *Cognitive Computation*, 16(4):1723–1734, February 2021. URL: http://dx.doi.org/10.1007/s12559-020-09801-w, doi:10.1007/s12559-020-09801-w.

[32] Agnieszka Truszkowska, Brandon Behring, Jalil Hasanyan, Lorenzo Zino, Sachit Butail, Emanuele Caroppo, Zhong-Ping Jiang, Alessandro Rizzo, and Maurizio Porfiri. High-resolution agent-based modeling of covid-19 spreading in a small town. *Advanced Theory and Simulations*, 4(3), January 2021. URL: http://dx.doi.org/10.1002/adts.202000277, doi:10.1002/adts.202000277.

[33] Dmytro Chumachenko, Ievgen Meniailov, Kseniia Bazilevych, Tetyana Chumachenko, and Sergiy Yakovlev. On intelligent agent-based simulation of covid-19 epidemic process in ukraine. *Procedia Computer Science*, 198:706–711, 2022. URL: http://dx.doi.org/10.1016/j.procs.2021.12.310, doi:10.1016/j.procs.2021.12.310.

[34] Gul Agha. *Actors: a model of concurrent computation in distributed systems*. MIT press, 1986.

[35] D. Kafura and J. Briot. Actors and agents. *IEEE Concurrency*, 6(2):24–29, April 1998. URL: http://dx.doi.org/10.1109/MCC.1998.678786, doi:10.1109/mcc.1998.678786.

[36] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[37] Myeong-Wuk Jang and Gul Agha. Agent framework services to reduce agent communication overhead in large-scale agent-based simulations. *Simulation Modelling Practice and Theory*, 14(6):679–694, August 2006. URL: http://dx.doi.org/10.1016/j.simpat.2005.10.002, doi:10.1016/j.simpat.2005.10.002.

[38] Matthias Scheutz, P Schermerhorn, R Connaughton, and Aaron Dingler. Swages-an extendable distributed experimentation system for large-scale agent-based alife simulations. *Proceedings of Artificial Life X*, pages 412–419, 2006.

[39] Ugo Erra, Bernardino Frola, Vittorio Scarano, and Iain Couzin. An efficient gpu implementation for large scale individual-based simulation of collective behavior. In *2009 International Workshop on High Performance Computational Systems Biology*, pages 51–58. IEEE, 2009.

[40] Michael J. North, Nicholson T. Collier, and Jerry R. Vos. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Transactions on Modeling and Computer Simulation*, 16(1):1–25, January 2006. URL: http://dx.doi.org/10.1145/1122012.1122013, doi:10.1145/1122012.1122013.

[41] Peter Wittek and Xavier Rubio-Campillo. Scalable agent-based modelling with cloud hpc resources for social simulations. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 355–362. IEEE, 2012.

[42] Nicholson Collier and Michael North. Parallel agent-based simulation with repast for high performance computing. *SIMULATION*, 89(10):1215–1235, November 2012. URL: http://dx.doi.org/10.1177/0037549712462620, doi:10.1177/0037549712462620.

[43] Lyndon Clarke, Ian Glendinning, and Rolf Hempel. The mpi message passing interface standard. In *Programming Environments for Massively Parallel Distributed Systems: Working Conference of the IFIP WG 10.3, April 25–29, 1994*, pages 213–218. Springer, 1994.

[44] Yuping Fan, Zhiling Lan, Taylor Childers, Paul Rich, William Allcock, and Michael E Papka. Deep reinforcement agent for scheduling in hpc. In *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 807–816. IEEE, 2021.

[45] Eduardo Felipe Zambom Santana, Nelson Lago, Fabio Kon, and Dejan S Milojicic. Interscsimulator: Large-scale traffic simulation in smart cities using erlang. In *Multi-Agent Based Simulation XVIII: International Workshop, MABS 2017, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 18*, pages 211–227. Springer, 2018.

[46] Actodes. `http://sowide.unipr.it/node/50`.

[47] Wayne Wolf. Cyber-physical systems. *Computer*, 42(03):88–89, 2009.

[48] Insup Lee and Oleg Sokolsky. Medical cyber physical systems. In *Proceedings of the 47th design automation conference*, pages 743–748, 2010.

[49] Zhicheng Fu, Zhao Wang, Chunhui Guo, Zhenyu Zhang, Shangping Ren, and Lui Sha. Iafinder: Identifying potential implicit assumptions to facilitate validation in medical cyber-physical system. In *Proceedings of the 55th Annual Design Automation Conference*, pages 1–6, 2018.

[50] Dimiter V Dimitrov. Medical internet of things and big data in healthcare. *Healthcare informatics research*, 22(3):156–163, 2016.

[51] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. The internet of things for health care: a comprehensive survey. *IEEE access*, 3:678–708, 2015.

[52] Rihab Chaâri, Fatma Ellouze, Anis Koubâa, Basit Qureshi, Nuno Pereira, Habib Youssef, and Eduardo Tovar. Cyber-physical systems clouds: A survey. *Computer Networks*, 108:260–278, 2016.

[53] Zhibo Pang, Junzhe Tian, and Qiang Chen. Intelligent packaging and intelligent medicine box for medication management towards the internet-of-things. In *16th international conference on advanced communication technology*, pages 352–360. IEEE, 2014.

[54] Nishat I Mowla, Inshil Doh, and Kijoon Chae. On-device ai-based cognitive detection of bio-modality spoofing in medical cyber physical system. *IEEE Access*, 7:2126–2137, 2018.

[55] Eric Xu, Marek Wermus, and Deborah Blythe Bauman. Development of an integrated medical supply information system. *Enterprise Information Systems*, 5(3):385–399, 2011.

[56] Roberto Martinez-Velazquez, Rogelio Gamez, and Abdulmotaleb El Saddik. Cardio twin: A digital twin of the human heart running on the edge. In *2019 IEEE international symposium on medical measurements and applications (MeMeA)*, pages 1–6. IEEE, 2019.

[57] Martin Wolfgang Lauer-Schmaltz, Philip Cash, John Paulin Hansen, and Anja Maier. Designing human digital twins for behaviour-changing therapy and rehabilitation: a systematic review. *Proceedings of the Design Society*, 2:1303–1312, 2022.

[58] Tina Hernandez-Boussard, Paul Macklin, Emily J Greenspan, Amy L Gryshuk, Eric Stahlberg, Tanveer Syeda-Mahmood, and Ilya Shmulevich. Digital twins for predictive oncology will be a paradigm shift for precision cancer care. *Nature medicine*, 27(12):2065–2066, 2021.

[59] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23, 2015.

[60] David De Roure, Kevin R Page, Petar Radanliev, and Max Van Kleek. Complex coupling in cyber-physical systems and the threats of fake data. In *Living in the internet of things (IoT 2019)*, pages 1–6. IET, 2019.

[61] Petar Radanliev, David C De Roure, Jason RC Nurse, Rafael Mantilla Montalvo, Stacy Cannady, Omar Santos, La'Treall Maddox, Peter Burnap, and Carsten Maple. Future developments in standardisation of cyber risk in the internet of things (iot). *SN Applied Sciences*, 2:1–16, 2020.

[62] Lyndon Clarke, Ian Glendinning, and Rolf Hempel. The MPI message passing interface standard. In *Programming Environments for Massively Parallel Distributed Systems*, pages 213–218. Birkhäuser Basel, 1994. doi:10.1007/978-3-0348-8534-8_21.

[63] Report real-time regione lombardia. https://experience.arcgis.com/experience/0a5dfcc103d0468bbb6b14e713ec1e30/, 2020.

[64] Github protezione civile. https://github.com/pcm-dpc/COVID-19, Real Time.

[65] Popolazione per età, sesso e stato civile 2020. https://www.tuttitalia.it/lombardia/statistiche/popolazione-eta-sesso-stato-civile-2020/, 2020.

[66] Comitato Tecnico Scientifico. Valutazione di politiche di riapertura utilizzando contatti sociali e rischio di esposizione professionale. https://www.quotidianosanita.it/allegati/allegato1389403.pdf, 2020.

[67] Popolazione per età, sesso e stato civile 2020. https://www.tuttitalia.it/lombardia/statistiche/popolazione-eta-sesso-stato-civile-2020/, 2020.

[68] Covid-19: nuovo studio sull'incubazione del sars-cov-2. https://www.osservatoriomalattierare.it/news/ricerca-scientifica/15771-covid-19-nuovo-studio-sull-incubazione-del-sars-cov-2, 2020.

[69] Epidemia covid-19. https://www.epicentro.iss.it/coronavirus/bollettino/Bollettino-sorveglianza-integrata-COVID-19_12-marzo-2020.pdf, 2020.

[70] Coronavirus, la demografia del lockdown: ecco l'identikit di chi lavora e di chi sta a casa. https://www.ilfattoquotidiano.it/2020/04/13/coronavirus-la-demografia-del-lockdown-ecco-lidentikit-di-\chi-lavora-e-di-chi-sta-a-casa/5763773/, 2020.

[71] Daniela Perrotta, André Grow, Francesco Rampazzo, Jorge Cimentada, Emanuele Del Fava, Sofia Gil-Clavel, and Emilio Zagheni. Behaviours and attitudes in response to the COVID-19 pandemic: Insights from a cross-national facebook survey. May 2020. doi:10.1101/2020.05.09.20096388.

[72] Steffen E. Eikenberry, Marina Mancuso, Enahoro Iboi, Tin Phan, Keenan Eikenberry, Yang Kuang, Eric Kostelich, and Abba B. Gumel. To mask or not to mask: Modeling the potential for face mask use by the general public to curtail the COVID-19 pandemic. *Infectious Disease Modelling*, 5:293–308, 2020. doi:10.1016/j.idm.2020.04.001.

[73] Albert-László Barabási. *Network Science*, chapter 4. The scale-free property and 10. Spreading Phenomena. Cambridge University Press, 2016.

[74] Powerlaw for python. https://pypi.org/project/powerlaw/, 2020.

[75] Seir and seirs models. https://docs.idmod.org/projects/emod-hiv/en/latest/model-seir.html.

[76] Giulio Zanella, Aldo Rustichini, Andrea Mattozzi, Andrea Ichino, Giacomo Calzolari, and Massimo Anelli. Transition steps to stop covid-19 without killing the world economy. https://voxeu.org/article/transition-steps-stop-covid-19-without-killing-world-economy, 2020.

[77] Studio della sieroprevalenza in italia. http://www.salute.gov.it/portale/news/p3_2_1_1_1.jsp?lingua=italiano&menu=notizie&p=dalministero&id=4998, 2020.

[78] Angelo Riccio. Analysis of the sars-cov-2 epidemic in lombardy (italy) in its early phase. are we going in the right direction? *medRxiv*, 2020.

[79] Alberto Godio, Francesca Pace, and Andrea Vergnano. Seir modeling of the italian epidemic of sars-cov-2 using computational swarm intelligence. *International Journal of Environmental Research and Public Health*, 17(10):3535, 2020.

[80] Yingcun Xia, Ottar N. Bjørnstad, and Bryan T. Grenfell. Measles metapopulation dynamics: A gravity model for epidemiological coupling and dynamics. *The American Naturalist*, 164(2):267–281, August 2004. URL: http://dx.doi.org/10.1086/422341, doi:10.1086/422341.

[81] Filippo Simini, Marta C. González, Amos Maritan, and Albert-László Barabási. A universal model for mobility and migration patterns. *Nature*, 484(7392):96–100, February 2012. URL: http://dx.doi.org/10.1038/nature10856, doi:10.1038/nature10856.

[82] Angelo Riccio. Analysis of the sars-cov-2 epidemic in lombardy (italy) in its early phase. are we going in the right direction? *medRxiv*, pages 2020–04, 2020.

[83] Covid-19 aggiornamento quotidiano dei dati-provincia reggio nell'emilia. https://www.ausl.re.it/covid-19-aggiornamento-quotidiano-dei-dati, 2022.

[84] Archivio github covid-19 dati regioni. https://github.com/pcm-dpc/COVID-19/tree/master/dati-regioni, 2022.

[85] Coordinate geografiche comuni italiani. https://www.dossier.net/utilities/coordinate-geografiche, 2022.

[86] Istat. https://www.istat.it, 2022.

[87] Analisi pendolarismo comune per comune della regione emilia romagna. https://statistica.regione.emilia-romagna.it/servizi-online/rappresentazioni-cartografiche/pendolarismo/analisi-comune, 2022.

[88] Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *sensors*, 12(9):11734–11753, 2012.

[89] RR Cima, MJ Brown, JR Hebl, R Moore, JC Rogers, A Kollengode, and Surgical Process Improvement Team, Mayo Clinic, Rochester. Use of lean and six sigma methodology to improve operating room efficiency in a high-volume tertiary-care academic medical center. *Journal of the American College of Surgeons*, 213(1):83–92, 2011. doi:10.1016/j.jamcollsurg.2011.02.009.

[90] N. Bahou, C. Fenwick, G. Anderson, R. van der Meer, and T. Vassalos. Modeling the critical care pathway for cardiothoracic surgery. *Health Care Manag Sci*, 21(2):192–203, 2018. doi:10.1007/s10729-017-9401-y.

[91] Y. Liu, M. Traskin, S. A. Lorch, E. I. George, and D. Small. Ensemble of trees approaches to risk adjustment for evaluating a hospital's performance. *Health Care Manag Sci*, 18(1):58–66, 2015. doi:10.1007/s10729-014-9272-4.

[92] S. A. Izad Shenas, B. Raahemi, M. Hossein Tekieh, and C. Kuziemsky. Identifying high-cost patients using data mining techniques and a small set of

non-trivial attributes. *Comput Biol Med*, 53:9–18, 2014. doi:10.1016/j.compbiomed.2014.07.005.

[93] D. Bertsimas, M. V. Bjarnadóttir, M. A. Kane, J. C. Kryder, R. Pandey, and S. et al. Vempala. Algorithmic prediction of health-care costs. *Operat Res*, 56(6):1382–1392, 2008. doi:10.1287/opre.1080.0619.

[94] B. Abbou, O. Tal, G. Frenkel, R. Rubin, and N. Rappoport. Optimizing operation room utilization—a prediction model. *Big Data Cogn Comput*, 6(3):76, 2022. doi:10.3390/bdcc6030076.

[95] N. Miyaguchi, K. Takeuchi, H. Kashima, M. Morita, and H. Morimatsu. Predicting anesthetic infusion events using machine learning. *Sci Rep*, 11(1):23648, 2021. doi:10.1038/s41598-021-03112-2.

[96] J. Futoma, J. Morris, and J. Lucas. A comparison of models for predicting early hospital readmissions. *J Biomed Inform*, 56:229–238, 2015. doi:10.1016/j.jbi.2015.05.016.

[97] B. Zheng, J. Zhang, S. W. Yoon, S. S. Lam, M. Khasawneh, and S. Poranki. Predictive modeling of hospital readmissions using metaheuristics and data mining. *Expert Syst Appl*, 42(20):7110–7120, 2015. doi:10.1016/j.eswa.2015.04.066.

[98] A. Bishara, A. Wong, L. Wang, M. Chopra, W. Fan, and A. et al. Lin. Opal: an implementation science tool for machine learning clinical decision support in anesthesia. *J Clin Monit Comput*, 36(5):1367–1377, 2022. doi:10.1007/s10877-021-00774-1.

[99] H. Hassanzadeh, J. Boyle, S. Khanna, B. Biki, and F. Syed. Daily surgery caseload prediction: towards improving operating theatre efficiency. *BMC Med Inform Decis Mak*, 22(1):151, 2022. doi:10.1186/s12911-022-01893-8.

[100] Igor I Immoreev and PGS Dmitry V Fedotov. Ultra wideband radar systems: advantages and disadvantages. In *2002 IEEE Conference on Ultra Wideband Systems and Technologies (IEEE Cat. No. 02EX580)*, pages 201–205. IEEE, 2002.

[101] Chris M Roberts. Radio frequency identification (rfid). *Computers & security*, 25(1):18–26, 2006.

[102] Raspberry pi 4 model b. `https://www.raspberrypi.com/products/raspberry-pi-4-model-b/`, 2024.

[103] Purnima Bholowalia and Arvind Kumar. Ebk-means: A clustering technique based on elbow method and k-means in wsn. *International Journal of Computer Applications*, 105(9), 2014.

Acknowledgements

The purpose of this thesis is to mark the end of a long and demanding journey, one filled with obstacles that I could overcome with the much-appreciated help of many people I was fortunate enough to meet along the way.

I want to thank my advisor, Professor Agostino Poggi, who joined me on this path with all the necessary support, guiding me with his experience in the field of this academic world that was completely new to me.

I would also like to thank my co-advisor, Prof. Gianfranco Lombardo. We shared, side by side, both the difficulties and the beauties of the research. In addition to finding an excellent colleague, I found an excellent friend. I am grateful to him for contributing to the researcher and the person I am today.

My gratitude goes also to the other members of my lab: Professor Monica Mordonini, Professor Stefano Cagnoni, and Professor Michele Tomaiuolo. Each of you supported and guided me in this new experience. Thank you for the trust you gave me, and thank you for giving me the possibility to acquire some knowledge and capabilities that I didn't have at the beginning.

The decision to be so far away from home and choosing a path of Ph.D. itself wasn't an easy choice. I had to make some tough choices and also go through obstacles that sometimes felt insurmountable. Luckily for me, I had the right people beside me who helped me through these tough times by standing with me. And for this reason I wouldn't be myself without my friends, too. Thanks to them, from Salento as well as from Parma. Mentioning everybody would be long and useless, but just know that, in case your name is not written in bold here, these lines hold all my gratitude,

admiration, and every other positive feeling that this small human being has for you.

Lastly, I would like to thank my family. This thesis is all for you. Anybody who knows me well can tell just how much my family means to me; it's the core of my universe. If I had not had your support, I would be like an astronaut lost in space without any destination. Still, you have always helped me by correcting my path when I seemed to be drifting off track. Thank you for having made my life so beautiful to live and enjoy. Many thanks, in particular, to those of you who left us too early and cannot see the completion of this work, but I'm sure you are following me from above with that smile of yours that I miss so much.

* * *

Questo lavoro di tesi rappresenta il completamento e la conclusione di un lungo percorso, ricco di sfide che sono riuscito a superare grazie al supporto di molte persone che, fortunatamente, ho incontrato lungo il cammino.

Vorrei ringraziare innanzitutto il mio relatore, il Prof. Agostino Poggi, per avermi accompagnato in questo percorso, fornendomi tutto il supporto necessario e guidandomi con la sua esperienza in un mondo accademico per me nuovo.

Un ringraziamento speciale va al mio co-relatore, il Prof. Gianfranco Lombardo. Abbiamo lavorato fianco a fianco e condiviso gioie e difficoltà nella ricerca. Oltre a trovare un eccellente collega, ho trovato anche un grande amico. Lo ringrazio per aver contribuito a formare il ricercatore e la persona che sono oggi.

Un particolare ringraziamento va anche a tutti gli altri membri del mio laboratorio: la Prof.ssa Monica Mordonini, il Prof. Stefano Cagnoni e il Prof. Michele Tomaiuolo. Tutti voi mi avete sostenuto e accompagnato in questa nuova avventura. Vi ringrazio per la fiducia che avete riposto in me e per avermi permesso di acquisire conoscenze e competenze di cui all'inizio ero privo.

Vivere lontano da casa e intraprendere la strada del dottorato non è stata una scelta facile. Mi sono trovato di fronte a decisioni difficili e ostacoli che, a volte, sembravano insormontabili. Fortunatamente, ho avuto accanto le persone giuste, che mi hanno aiutato a superare questi momenti difficili stando al mio fianco nei periodi più bui. Ed è per questo che affermo che non sarei me stesso senza i miei amici,

che desidero ringraziare tutti, sia quelli Salentini sia quelli Parmigiani (e Parmensi). Fare un elenco di nomi sarebbe lungo e superfluo, ma sappiate che, anche se il vostro nome non è scritto nero su bianco, queste righe racchiudono tutta la mia gratitudine, ammirazione e ogni altro sentimento positivo che questo piccolo essere umano prova per voi.

Infine, vorrei ringraziare la mia famiglia. Questa tesi è dedicata interamente a voi. Chi mi conosce sa quanto io tenga alla mia famiglia, che è il centro del mio universo. Senza il vostro supporto sarei solo un astronauta smarrito, destinato a vagare nello spazio senza meta. Invece, voi mi avete sempre aiutato e corretto la rotta quando sembrava che stessi uscendo fuori dal percorso. Grazie per aver reso la mia vita così bella da vivere e da godere. Un ringraziamento speciale va anche a chi mi ha lasciato troppo presto e non può vedere questo lavoro compiuto, ma sono certo che mi guardiate con il vostro sorriso, quello che tanto mi manca.