School of Electronic Engineering and Computer Science
Queen Mary University of London

FINAL YEAR PROJECT DEFINITION
2024-25

This project definition must be undertaken in consultation with your supervisor. The feasibility of the project should have been assessed and the project aims should be clearly defined.
Submission of this document implies that you have discussed the specification with your supervisor.

Project Title: Software for Simulating Moran Process with Multiple Mutations

Supervisor: Dr. Marc Roth

Student name: Thomas Louis Sigone

Student e-mail: t.l.sigone@se22.qmul.ac.uk

BACKGROUND: *Relevant research, an overview of the need for the project, related existing systems, a gap you have identified.*

The Moran Process is a fundamental model in theoretical computer science used to study how mutations spread within a population. While extensively studied with single mutations, scenarios involving multiple mutations are less explored despite their relevance in real-world applications such as genetics, evolutionary biology and Complex Network Theory.
In the multi-type Moran process, each vertex on a connected graph represents one of several mutation types, and at each step, a vertex reproduces its type to a neighbour with a probability proportional to the fitness of that type. Much of the earlier research has focused on the two-type Moran process - where vertices are either healthy or mutated, with the fixation probability measuring the likelihood that all vertices eventually become mutants. However, recent research by Goldberg, Roth, and Schwarz [1] has introduced a fixed-parameter tractable randomized approximation scheme (FPTRAS) for estimating fixation probabilities in multi-type Moran processes, considering various mutation types and their fitness levels.
Despite these advances, existing software solutions are limited to simpler cases and do not support simulations involving multiple competing mutations or real-world data integration. EvoLudoLab, for instance, provides simulations of the Moran process on random regular graphs and allows users to study evolutionary dynamics on structured graphs, but it lacks features for handling complex multi-type mutation scenarios or automated large-scale graph modifications [2]. This project aims to fill that gap by

developing flexible software capable of simulating the multi-type Moran process, supporting both manual inputs and dataset imports, automating graph modifications via scripts, and calculating mutation fixation probabilities.

PROJECT AIMS:
State the design, development or research challenge (problem) that the project aims to solve.

The project aims to develop a software tool that simulates the Moran Process with multiple mutations and calculates the probability of one mutation taking over the population. It will allow users to input data manually or import datasets, offering a detailed analysis of mutation dynamics under various conditions, such as mutation rates and population size. A key component of this project will be the implementation of a user-friendly Graphical User Interface (GUI), which may be available as either a desktop, an online application, or both; enabling users to interact with the simulation intuitively. Moreover, the project addresses a mathematical problem that can represent an abstraction of real-world applications and scenarios, such as the spread of diseases, the evolution of species, and the spread of misinformation over the internet. This mathematical foundation allows for flexible applications across various fields, offering valuable insights into population dynamics and mutation spread in both theoretical and practical contexts.

PROJECT OBJECTIVES:
List a series of objectives you need to achieve to fulfil the aims of your project.

1. **Research & Literature Review:** conduct a comprehensive review of existing models of the Moran process and multi-mutation scenarios.
2. **Simulation & Algorithm Development:** implement algorithms to simulate the Moran process with multiple mutations.
3. **Data Analysis & Visualization:** calculate and visualize the probability of different mutations overtaking the population.
4. **Software Development and GUI:** develop an graphical user interface for both manual input and dataset imports.
5. **Testing & Experimentation:** test the software using both synthetic and real-world datasets to experiment with different conditions and validate the overall functionality and performance of the simulation.
6. **User Experience:** ensure the software is user-friendly for researchers and students, and includes comprehensive documentation.
7. **Exploring applications:** explore how the mathematical model of the Moran Process can represent an abstraction of real-world applications, implementing a few example scenarios.

State how your project will be aligned with the learning outcomes of your programme of study.

This project aligns with the learning outcomes of my programme by integrating concepts from theoretical Computer Science, algorithm development, software engineering, and probability theory, providing a holistic approach to solving a complex research challenge

METHODOLOGY:

Describe the various steps that you intend to follow to achieve your project aims.

- **Literature Review:** conduct an in-depth review of existing research on the Moran Process, with a particular focus on multi-mutation models and recent advancements in computing fixation probabilities. This will provide the theoretical foundation for the project and help identify deficits in existing tools.
- **Software Development:**
    - **Programming Languages**, such as Python or Java, depending on compatibility and performance needs. Python is preferred for its compatibility and valid libraries in data science.
    - **Libraries**, such as NumPy for efficient numerical computation, Pandas for data handling, and possibly Matplotlib for data visualization.
    - Web **frameworks** like Django (for an online GUI) or desktop application frameworks such as PyQt or Tkinter (for a desktop version) to build the user-friendly interface.
- **Interface Design**: design a GUI that allows users to either input data manually or import external datasets. The interface should be intuitive and support real-time simulation control, ensuring ease of use for both non-experts and advanced researchers.
- **Algorithm Implementation:** implement efficient algorithms to simulate multiple competing mutations and compute their probabilities. Adapt from existing literature on computational models of evolutionary dynamics, such as those provided by Goldberg and Roth's FPTRAS algorithm for multi-type mutation scenarios [1].Ensure that the algorithms are optimized for scalability and can handle both small and large graphs efficiently.
- **Testing & Experimentation:** test the software across various scenarios, starting with simple, controlled simulations (e.g., small population sizes or limited mutations) and then moving on to larger, real-world datasets (e.g., genetic data or network data). Experiment with different mutation rates, population structures, and fitness advantages to ensure the validity of the software under various conditions.
- **Data Analysis & Visualization:** develop data analysis tools to process simulation outcomes, such as calculating fixation probabilities and tracking mutation dynamics over time**.** Implement advanced visualization features (e.g., heatmaps, probability graphs) to provide users with clear insights into the dynamics of mutations within the simulated population. Ensure that the visual outputs are adaptable for different types of datasets and are accessible within the GUI.

- **User Feedback & Refinement:** gather feedback from users (e.g., students, researchers and lecturers) to refine the interface, simulation accuracy, and user experience. The feedback will be essential for improving the software and ensuring that it meets the needs of its target audience.

PROJECT MILESTONES

Indicate what measurable/tangible components you will produce as part of this project. This may take the form of deliverable document(s) or developmental milestones such as a working piece of software/hardware.

1. **Literature Review Completed.**
   Reseach on existing models on the Moran process, multi-type mutations, and related simulation techniques.
   Deliverable: key findings and gaps to address.
2. **Research and Setup of Programming Environment.**
   Research and choose the appropriate programming languages, libraries, and frameworks (e.g., Java, Python, JavaFX, NumPy, etc.). Set up the development environment, ensuring all necessary tools are installed and configured.
   Deliverable: development environment fully set up, with project dependencies and libraries installed, and initial project repository created.
3. **Algorithms for Simulation Implemented.**
   Complete the development of core algorithms for simulating the multi-type Moran process, including the computation of fixation probabilities.
   Deliverable: fully functioning algorithm module that handles mutation simulations.
4. **Software Interface Designed.**
   Finalize the design of the graphical user interface (GUI), including wireframes or mockups that specify functionality for manual inputs and dataset imports.
   Deliverable: GUI fullly functioned and satisfactory design.
5. **Initial Testing with Datasets.**
   Test the software with synthetic datasets to evaluate functionality and identify any performance issues.
   Deliverable: testing report with results, identifying bugs or limitations found during testing.
6. **Dataset Import Functionality Developed.**
   Implement functionality that allows users to import external datasets for simulations.
   Deliverable: dataset import module integrated into the system and demonstrated using sample datasets.
7. **Applications Demonstration Completed.**
   Develop and showcase example scenarios where the software models real-world applications (e.g., cancer evolution, viral mutations).
   Deliverable: at least two real-world applications simulated by the software.
8. **Final Software Testing & Validation.**

Perform comprehensive testing with real-world datasets and scenarios to validate accuracy, performance, and reliability.
Deliverable: final testing and validation report, detailing the accuracy of the simulation under different conditions.

9. **Documentation and User Guide Finalised.**
   Complete all documentation, including a user manual, technical documentation, and final project report.
   Deliverable: Full documentation package, including user guide, technical details, and any setup instructions.

REQUIRED KNOWLEDGE/ SKILLS/TOOLS/RESOURCES:

Indicate as far as possible the skills that are required for you to undertake this project. Also include any software, hardware or other tools or resources that you believe you will need.

- **Skills:**
  - Understanding the Moran Process and its applcations.
  - Software development and algorithms knowledge.
  - Data visualization techniques.
  - Basic probability theory.

- **Tools:**
  - Programming languages: Python or Java.
  - Libraries: NumPy (for mathematical modeling), Matplotlib (for visualization), Pandas (for dataset manipulation).
  - Frameworks: Django, PyQT or JavaFX.
  - IDE: PyCharm, IntelliJ IDEA or Visual Studio.
  - Version Control: Git and Github.
  - Dataset integration tools.

TIMEPLAN

This can be a GANTT chart submitted with this document or a list of tasks, milestones and deliverables with timings.

- Week 1 – 4:
  Literature Review.
- Week 2 – 5:
  Research and Setup of Programming Environment.
- Week 4 – 14:
  Algorithms for Simulation Implemented.
- Week 8 – 16:
  Software Interface Designed.
- Week 9 – 17:
  Initial Testing with Dataset.
- Week 10 – 18:
  Dataset Import Functionality Developed
- Week 13 – 20:
  Applications Demonstration Completed.
- Week 20 – 24:
  Final Software Testing & Validation.
  Documentation and User Guide Finalised.

REFERENCES

[1] L. A. Goldberg, M. Roth, and T. C. Schwarz, "Approximation of Fixation Probabilities in the Multi-Type Moran Process," *Theoretical Computer Science*, vol. 984, 2024. Available at: https://www.sciencedirect.com/science/article/pii/S030439752400402X.
[2] A. Traulsen, "EvoLudoLab: Moran Process on Random Regular Graphs," EvoLudo Project. Available at: https://wiki.evoludo.org/index.php?title=EvoLudoLab:_Moran_process_on_random_regular_graphs.