Data driven agent-based micro-simulation in social complex systems

DATA DRIVEN AGENT-BASED MICROSIMULATION IN SOCIAL COMPLEX SYSTEMS

O.A MAKINDE

PHD

2019

DATA DRIVEN AGENT-BASED MICRO-SIMULATION IN SOCIAL COMPLEX SYSTEMS

Omololu Ayodeji MAKINDE

A thesis submitted for the degree of

Doctor of Philosophy

Department of Computer Science

University of Bradford

2019

Omololu A Makinde

DATA-DRIVEN AGENT BASED MICRO SIMULATION FOR POLICY EFFECT PREDICTION IN SOCIAL COMPLEX SYSTEMS

Keywords: Social complex system, Micro-simulation models, Agent-based models

Abstract

We are recently witnessing an increase in large-scale micro/individual/-granular level behavioural data. Such data has been proven to have the capacity to aid the development of more accurate simulations that will effectively predict the behaviours of complex systems. Despite this increase, the literature has failed to produce a structured modelling approach that will effectively take advantage of such granular data, in modelling complex systems that involve social phenomenons (i.e. social complex systems).

In this thesis, we intend to bridge this gap by answering the question of how novel structural frameworks, that systematically guides the use of micro-level behaviour and attribute data, directly extracted from the basic entities within a social complex system can be created. These frameworks should involve the systematic processes of using such data to directly model agent attributes, and to create agent behaviour rules, that will directly represent the unique micro entities from which the data was extracted. The objective of the thesis is to define generic frameworks, that

would create agent based micro simulations that would directly reflect the target complex system, so that alternative scenarios, that cannot be investigated in the real system, and social policies that need to be investigated before being applied on the social system can be explored.

In answering this question, we take advantage of the pros of other modeling techniques such as micro simulation and agent based techniques in creating models that have a micro-macro link, such that the micro behaviour that causes the macro emergence at the simulation's global level can be easily investigated. which is a huge advantage in policy testing. We also utilized machine learning in the creation of behavioural rules.This created agent behaviours that were empirically defined. Therefore, this thesis also answers the question of how such structural framework will empirically create agent behaviour rules through machine learning algorithms.

In this thesis we proposed two novel frameworks for the creation of more accurate simulations. The concepts within these frameworks were proved using case studies, in which these case studies where from different social complex systems, so as to prove the generic nature of the proposed frameworks.

In concluding of this thesis, it was obvious that the questions posed in the first chapter had been answered. The generic frameworks had been created, which bridged the existing gap in the creation of accurate models from the presently available granular attribute and behavioral data, allowing the simulations created from these models accurately reflect their target social complex systems from which the data was extracted from.

Declaration

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

Omololu A Makinde

Acknowledgements

I would like to thank my God, the almighty, whose mercies have endured. I would also like to thank my family, especially my parents, Mr and Mrs Makinde, for their love and support. I also like to acknowledge all my friends, they are all too many to mention, and finally i want to say a special thank you to may two supervisors, Prof. Daniel Neagu and Prof. Marian Gheorghe, who never gave up on me in the days of my folly.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the last few decades, micro-simulation models (MSM) have been successful in modelling social complex systems (SCS), especially in the testing and development of public policies. The strength of micro-simulation models in replicating complex policy structures gives them the capability to forecast the outcome of policy changes and "what-if" scenarios. A social policy, according to [73], and in the context of this thesis, refers to a course or principle of action adopted or proposed on the social system being observed. "What-if" scenarios, is a term that refers to alternate scenarios that are too expensive to be tested on the actual system, and are therefore being analysed on the simulation to predict their effect [36].

However, this strength fosters a weakness which reveals itself in the fact that these models only model one direction interactions such that even though it can explore the impact of the policy on its smallest individual unit, it is weak in exploring the impact the individual unit has on the policy, causing them to be less strong in the area of behavioural modelling, as these individual units do not interact.

Although micro-simulations are data and computational intensive, this is no longer

a challenge due to the availability of high quality individual level data and the high level computation capability available today. But yet, the robustness of the behavioural basis to these simulation models can still be questioned due to the fact that the units within a micro-simulation do not model the interactions between the real individuals in the target system. Hence, the model is unable to justify the behaviours of each individual in terms of its individual preference, decision, plans, etc, which altogether impair the models validation.

Agent based models (ABM) on the other hand is also suited for simulating social science phenomena, based on the fact that it excels in simulating situations where there is a large number of heterogeneous individuals who may behave differently. Since these individual entities within the target system are represented by autonomous agents in the model, there is a capacity for these agents to communicate and interact, just as the entities of the real system would. This allows the exploration of such interactions at a local level and also at a global level (Such global level interaction will be an emerging behaviour of the whole model, resulting from the combination of all the local interactions and individual agent behaviours).

The philosophy of the ABM is rooted in the idea that the interaction between these autonomous agents can produce the behaviour which will reflect the aggregate sum of the individual activities of the reflected entities in the target system. Therefore the strength of the ABM is in modelling the behavioural attributes and environment of the individual entities of the target social system to create agents whose interactions are in turn explored, in order to analyse, understand and explain the target system.

Yet the ABM has its setbacks, in that most interactions of ABM are set and calibrated according to the target systems social norms, settings and assumptions (assumptions that reflect the social context and previous research results), hence requiring less

usage of real individualistic empirical data. Also, many applications of ABM to public or social policy domains involve the development of alternative scenarios to facilitate decision making. These in itself pose a challenge to validation and policy testing.

1.1 Agent Based Micro Simulations

Micro simulation models (MSM) and agent based models (ABM) have some similar properties, such as accommodating heterogeneous entities, leveraging a bottom up approach and modelling individual entities on a micro-level. They also have some characteristics that distinguish them, such as: the heavy use of individual level empirical data by MSM as compared to conceptual rules by ABM, individual interactions dominant in ABM but absent in MSM, the probabilistic approach to an agent's state change used by MSM versus the rule based approach to an agent's state change, embraced by ABM, and while MSM is focused on predicting policy effects in the simulated system, ABM is focused on replicating the interactions within the simulated social system, so as to produce an emergent behaviour similar to the general behaviour of the system.

Therefore, in an effort to produce more realistic models, we take advantage of the strengths of both models while mitigating there weaknesses. To achieve this, we need to develop frameworks that amalgamate these two techniques. Such agent based micro simulation (ABMS) should be able to:

- Predict and analyse the transition link from local behaviours and interactions to global outcomes.
- Achieve consistency with the world outside a defined core system boundary by merging empirical data with conceptual rules.

- Simultaneously represent processes on different spatial and temporal scales.
- Integrate observable and conceptual behaviours while retaining the ability of the model to achieve endogenous emergence.

One way of creating such hybrid models is to create a data-driven modelling approach that produces the behaviour and environment of the agents directly from empirical real life data. This will allow the micro simulation the capacity not just to explore the impact of policies on an individual in the target system, but will shed light on how each individual's decision making process influences the impact of the policy.

Also, by generating agent behaviours, environment rules and by initializing agent attribute values using empirical individual level data, one can create a model that can be easily validated with empirical data. And with empirical data defining a good part of the agent behaviour, a stochastic approach to state change can be ushered in to create more realistic models.

1.2 Towards A Data-Driven Agent Based Micro Simulation

Micro simulation models (MSM) for social systems use micro-level individual data to initialize the primary basic unit of the target system so as to produce a model such that when simulated will produce an aggregate output that can be used in testing policies. This is different from an ordinary simulation model that does the same thing except that it uses aggregated data instead of individual-level data to initialize the primary basic unit of the target system.

On the other hand, agent based models (ABM) create autonomous agents whose

behaviours are replicas of the primary basic units of the target systems, such that these agents interact together to create an emergent global behaviour.

But, with the proliferation of large scale behavioural data at micro individual level, and the advancement of big data techniques and data mining tools, it has now become possible to create an agent based micro simulation (ABMS) whereby individual level empirical data can be used to generate agent behavioural rules and initialize agent attributes.

This approach has been used in creating a number of models, but the literature has not produced structured approaches to produce agents directly from data, especially dynamic data. Therefore, we propose initial steps towards a data-driven agent based micro simulation framework that will focus on using static and dynamic individual level data to generate agent behavioural rules and initialize agent attribute values.

1.3 Scope

The scope of this thesis is limited to proposing generic frameworks for modelling complex systems characterised by social interactions (i.e. social complex system).

Within the scope of this thesis, we refer to an agent as an independent entity with the ability to pursue a goal. A social agent is an agent in the human system, such a people, nations e.t.c. From the view point of this thesis, a complex system is regarded as a system whose behaviour cannot be easily predicted from inspecting the system [84]. A social complex system is a complex system whose behaviour is primarily the result of the behaviour of social entities. Example include families, nations and online users [84].The complex systems dealt with and refereed to in this thesis is limited to only social complex systems and not physical or natural complex systems or otherwise

[54].

Also, a social policy, according to [73], and in the context of this thesis, refers to a course or principle of action adopted or proposed on the social system being observed. A typical example is a new pricing policy adopted or introduced by a public transport organization (PTO). “What-if” scenario, is a concept that refers to alternate scenarios that are too expensive to be tested on the actual system, and are therefore being analysed on the simulation to predict their effect [36]. Therefore if a public transport organization (PTO) is investigating a new pricing policy, the management team will want to understand the “What-if” scenarios for every strategy under consideration, before adopting one.

This thesis uses data and autonomous agents as the main building blocks in simulating the afore mention systems. These frameworks utilize the principles of both ABM and MSM in creating models that are useful in analysing policies and ’what-if’ scenarios that apply to the complex system being simulated. These frameworks approach the creation of models using a bottom up approach, such that, the smallest entities and their environments, within the chosen social system are identified, studied and recreated into autonomous agents. In this research, the smallest entities in the social complex system are regarded as:

- The smallest unique entity from which behaviour and attribute data can be extracted from. For example, the passenger rail simulation discussed in chapter 4 regards every individual passenger and every individual train as the smallest entities. Note that passengers are not the smallest element, neither are trains. An individual passenger or an individual train is the smallest entity, because it is the smallest entity possible to extract travel behaviour and personal attributes from each passenger, and the same goes for each train. Also, in the computer network

simulation of chapter 5, we regarded the smallest elements as each individual network user and each individual computer work station, because they are the smallest elements from which we are able to extract individual behaviour and attribute data from each user and each workstation.

- The entity discussed above must be able to interact with its environment and other entities within its environment.

- Such environment should have the possibility of being modelled, using data extracted from the environment and/or using theoretical facts about such environment or alternative environments.

Since the framework deals with individual level entities in creating agents, the data sets used in this thesis contain individual level or micro level data, as opposed to aggregate level data. These datasets where extracted directly from the modelled complex system.

The case study models used in this thesis where scaled down, since we were unable to obtain data on every single individual entity in the systems chosen.

Although, this research encourages the use of data in creating models of the entities within the complex system, the modelling of these entities is not restricted to using data only data. The frame work proposes a part use (if necessary) of theoretical concepts in creating the models.

1.4 Research Questions

We have recently witnessed an increase in large-scale micro/individual level, behavioural data that can be used to empirically develop agent-based models (ABMs). Micro level

data, also used synonymously as granular or individual level data in this thesis refers to data extracted from the lowest level entity in a social system. A typical example is the historical blogs of a specific user of a target social media platform, travel movement of a specific passenger of a chosen transport system e.t.c. The usefulness of such data is in its potential to shed light on the unique characteristics and behaviour of such micro entity which would in turn enable the creation of micro agents so as to aid a bottom up approach in building the target social complex systems.

Despite the increase in such granular data, the literature has failed to produce a structured agent-based modelling framework to produce granular agents or its parts directly from such data. It is based on this that the following research question is asked.

- Research Question 1 (RQ1): How can a framework for modelling social complex system be created, such that:

  1. the architecture will be generic enough to be applied in modelling any social complex system,
  2. micro level static data extracted from the smallest entity in the complex system will be used in initializing the corresponding agent attributes and its environment,
  3. micro level static data extracted from the smallest entity in the complex system will be used in creating the behaviour rules of the corresponding agent of the entity,
  4. the overall simulation be easily validated using empirical data extracted from the target system?

Also,based on the fact that human behaviours are subject to change over time, and the environment around the human subject in a social system can also change or can instigate behavioural change, this phenomena will annul the attributes and behavioural patterns extracted from the historical data previously gathered. Therefore, it becomes obvious that a model created from static data will have to be consistently updated with new data to keep it continually accurate. This updating process has to be done, such that the new data permeates through the whole simulation and is not just used as a simple parameter tuner.

Based on this fact, we ask a second question:

- Research Question 2 (RQ2): How can a framework for modelling social complex system be created, that will not only answer research question one above, but also be up-datable, such that:

  1. micro level dynamic data extracted from the smallest entity in the complex system will be used in initializing the corresponding agent attributes and its environment,
  2. micro level dynamic data extracted from the smallest entity in the complex system will be used in creating the behaviour rules of the corresponding agent of the entity,
  3. the overall simulation be easily validated using empirical data extracted from the target system?

1.5 Research Approach

The strategy used in this thesis is a mix of exploratory and empirical research with case studies used as proof of concepts in backing up proposed concepts. The methodology adopted is in line with the research process used in (Leskovec 2008), which involves observation, model design and algorithm/system development.

Firstly, a review of recent literature on the different techniques applied in this research is explored, including other frameworks that have sought to answer the research questions asked in the previous section. This review is done with the intention of highlighting the research gaps in this area of interest. Furthermore, innovative frameworks for data-driven agent based micro simulations have been proposed, and the overall goal of developing these is to create agents from individual level data, such that, these frameworks will be generic enough to allow the creation of more realistic models that can be easily validated.

Case studies are then used to test the efficacy of the proposed frameworks. Target systems that involve human interactions and behaviour prediction have been used as case studies, since they represent difficult modelling examples that will involve the usage of all aspects of the proposed frameworks.

In choosing the right case studies, the following considerations where made:

1. The presence of autonomous human interactions, which is the core of the type of complex system being considered in this thesis.

2. The availability of individual level human interaction data, such that the behavioural and attribute data of every human interaction within the complex system is available.

3. The availability of the environment data in which the human entity is operating.

Two complex systems that easily fulfill these considerations, are the transportation system and the computer network system.

In a typical transportation system (which could be car, rail, airline e.t.c.), passenger interactions, both within themselves and with the environment is core to the functioning of the complex system. Such interactions are autonomous, and the data trail of passenger decisions can be easily recorded through various means such as tickets and surveys. The environment data, which could the mode of transport, stops, and route, can easily be gotten from the public transport organization (PTO). Also, in a typical computer network system, users interact autonomously with themselves and with the computer network. The attributes and behaviours of these users can easily be captured using network logging details, online trailing software e.t.c. The environment data, which could be the available software on the network, security protocols e.t.c. can be easily gotten from the system administrator. Based on these enumerated facts, the following case studies have been chosen for this thesis:

The first social system case study involves a passenger rail transport system, which involved simulating the daily travel of passengers through a commuter rail. The data used in implementing this case study is a secondary data set gotten from a commercially active public transport organization (PTO). The second case study is a distributed computer network (DCN) request model, that focused on user behaviour within a computer network. The data for this model is a primary data set extracted directly from the computer network.

At the end of these case studies, our models behaved so much like the original complex systems, that we were able to recreate alternative (what-if) scenarios. For our transportation model, we created alternative scenarios involving different pricing

policies, thereby allowing the prediction of demand at such prices. For our distributed computer network (DCN) model, we created alternative scenarios that involved the intrusion of external virus soft wares, to show how the network can be compromised by vulnerable users.

The above case studies are used in order to validate the generic nature of the proposed framework. The intention is to prove that the framework is generic enough to be applicable in as many social complex systems as possible.

1.6 Contributions

The contributions of this thesis to the discipline of artificial intelligence and computer science include:

- A novel framework for data-driven agent based micro simulation with focus on static behavioural data at the individual agent level, to create behavioural rules and to initialize attributes, such that, the overall simulation could be validate using empirical data.
- A novel framework for data-driven agent based micro simulation with focus on dynamic behavioural data at the individual agent level to create behavioural rules and to initialize attributes, such that the overall simulation could be validate using empirical data.
- A novel test bed for testing the impact of customer reaction to price policy change of public transport system.
- A novel test bed for testing the vulnerability of distributed computer networks. This test bed, created from a model of the user requests within a chosen computer

network, is highly efficient in testing the vulnerability of a computer network to a viral attack.(This project falls into the scope of simulating social systems, because, the focus is on how the users of this computer network can determine how vulnerable the security of the network is, to a computer virus).

This thesis is based on the summary of the following academic publications as well as other new unpublished studies.

- **Agent Based Micro-Simulation of a Passenger Rail System Using Customer Survey Data and an Activity Based Approach:** In this paper we tackled the problem of passenger rail overcrowding, which is fast becoming a problem in major cities world-wide. This problem requires accurate modelling tools to effectively forecast the impact of transit demand management policies. To do this, we developed an agent-based model of a particular passenger rail system using an activity based simulation approach to predict the impact of public transport demand management pricing strategies. Our agent population was created using a customer/passenger mobility survey dataset. We modelled the temporal flexibility of passengers, based on patterns observed in the departure and arrival behaviour of real travellers. Our model was validated using real life passenger count data from the passenger rail transit company, after which we evaluated the use of peak demand management instruments such as ticketing fares strategies, to influence peak demand of a passenger rail transport system. Our results suggest that agent-based simulation is effective in predicting passenger behaviour for a transportation system, and can be used in predicting the impact of demand management policies. The paper presentation was made in August 2018 during the 18th UK Workshop on Computational Intelligence (UKCI) held at Notting-

ham Trent University.

- **Distributed network behaviour prediction using machine learning and agent-based micro-simulation:** In the past, most researchers have predicted network behaviour and network attack patterns by using aggregated data, but in this paper, we focus on the application of machine learning at the individual user level, such that the prediction of the individual network user behaviour pattern at the micro-level becomes a substantive tool in creating a realistic agent based simulation of the whole distributed network, which in turn can serve as a test bed for predicting what-if scenarios such as network attacks on the target system or exposing vulnerabilities within the target system. This paper was presented at the 7th IEEE International Conference on Future Internet of Things and Cloud, Turkey, July 2019

- **A Framework for Dynamic Data-Driven Agent Based Micro Simulation:** This contribution is currently being considered for publication in a journal. It featured the creation of a framework that will extend the research in data-driven agent based micro-simulation, which respects the granularity of this concept by seeking to update the individual agents with the consistently changing data (i.e, dynamic data) of their corresponding entities within the social system. This created a more accurate model for predicting policies and aiding decision making.

- **A Distributed Computer Network Using Framework for Dynamic Data Driven Agent-Based Micro-Simulation:** This paper is currently being considered for a journal publication. It features the creation of a dynamic data driven agent based micro simulation (DDDABMS) framework. This framework seeks to update individual agent's behaviour and attributes with the consistently changing

data (i.e, dynamic data) of the corresponding entity within the social system. A case study, involving the creation of a distributed computer network was applied in validating this concept. This model was able to maintain accuracy continuously through time, by evolving the agent's behaviour and attributes through he infusion of dynamic data into the simulation.;

1.7 Outline

The remainder of this thesis is structured into seven chapters, with some chapters building up to the contributions and the rest detailing their proof of concepts, and applications.

This chapter has introduced the problem context after which the research question, methodology and contributions have been stated. The subsequent chapters are:

- Chapter 2: provides a literature review, in which the state of art research development related to this study is discussed. The first and second sections introduces the concept of microsimulation and agent based modelling, respectively, including state of art models that have used these individual techniques. The third section talks about different solutions that have amalgamated both modelling approaches, and the last section gives a state of the art of the present frameworks that have attempted to structure the application of big data, data mining and machine learning in the amalgamation of agent based modelling and microsimulations, while characterizing the different data usages in these hybrids.

- Chapter 3: proposes a framework for data-driven agent based micro simulation using static data. In doing this, the different aspect of the framework is expanded

upon.

- Chapter 4: A model is developed using the framework proposed in chapter three, with the aim of proving the concept proposed in chapter three. A data-driven agent based micro simulations model of a passenger rail system is built. Real individual passenger, train, stops and track historic data was used, which was collected from a public transport company call Metro Transport Authority (MTA). The model is then further used as a decision making test-bed for predicting pricing policies for the public transport company.

- Chapter 5: Another model is developed using the framework in chapter three. This data-driven agent based micro simulation model is used in recreating a distributed computer network using historic data collected from individual workstations in the network and user on-line movement data. The developed model was further used in determining the networks vulnerability, with the intent of aiding security policies in the network.

- Chapter 6: proposes a framework for dynamic data-driven agent based micro simulation. This chapter highlights the adaptability shortcoming of the framework introduced in chapter three in the face of a constantly changing complex system. The proposal to solve this shortcoming is structured by creating a framework that can accommodate constantly changing data from the individual entities being modelled within the target complex system. The adaptability of this framework is demonstrated by recreating the distributed network model of Chapter 5 using the proposed framework and comparing it to the model created using the previous framework.

- Chapter 7: is a recapitulation of previous chapters and suggestion of directions for future research.

Chapter 2

Related Research

This chapter details the state of art research development related to the areas of agent based models (ABM), micro simulation models (MSM), big data and the different research works that have tried to use some or all of these technologies in simulating social complex systems.

Sections 2.1 and 2.2 introduce the micro-simulation and agent based modelling techniques. These sections also discuss the state of the art models that have used these individual techniques. Section 2.3 talks about individual level (i.e, micro level) big data, while section 2.4 discusses different solutions that have attempted to structure the application of data in hybrid models that include agent based modelling and micro simulations. This section also characterises the different data usages in these hybrids.

This kind of simulation is driven by transition probabilities, derived from real life data, which is used in determining the change and progression in the simulation. Although, it is data intensive and is characterised by micro-level data, its major feature is policy relevance, such that, “what-if” scenarios can be simulated, and the impact of new policy rules on individual micro-units and subgroups, or on the whole system can

be analysed.

2.1 Micro Simulation Modelling Technique

MSM Technique captures the detailed interaction between policy and social economic behaviours of people by simulating distinctive behaviours and characteristic at the level of individual decision making units [80]. It describes a target system at the individual level, such that the system dynamic is represented as the aggregation of individuals.

This kind of simulation is driven by transition probabilities, derived from real life data, which is used in determining the change and progression in the simulation. Although it is data intensive and characterised by micro level data, its major feature is policy relevance, such that "what-if" scenarios can be simulated in which the impact of new policy rules on individual micro units and subgroups or on the whole system can be accessed.

A typical MSM starts with an entity population $E = [e_1, e_2, .., e_n]$, where n is the total number of the individuals in the population sample. Each individual has a set of unique attributes $A = [a_1, a_2, .., a_m]$ describing them at time t where m is the total number of features. This creates an $n \times m$ array of entities and attributes that can then be populated using reliable data or estimates from empirical data such as survey data sets. The array is then updated so that the baseline population $[e_1A, e_2A, .., e_nA]$ changes to new sets with states at points: $t + 1, t + 2..$ and so on.

Since MSM is based on unit records, it allows the evaluation of the impact of policies on individual decision units. This distinguishes it from traditional mathematical models which are often based on aggregated or averaged values which could eventually blur individual characteristics.

MSM as a technique was pioneered by the works of Guy Orcutt [80]. His research application was directed towards income modelling which then resulted in the creation of DYNASIM (Dynamic Simulation of Income Model) [30], which was designed for analysing the distributional consequence of retirement ageing issues. Although it has been revised over the years, DYNASIM was created using a self-weighting sample data of 130,000 people and 46,000 families, serving as the main micro level input data, which was based on the 1990 to 1999 survey of income and programme participation panels (SIPP) in the united states if America (USA). This MSM has been used in evaluating the impact of different policies on the income of retirees [91][16][29].

Ocutt's model also inspired the creation of CORSIM (Cornell micro simulation model), which aims at modelling large scale government programs, especially social security programs. CORSIM's data input is also a micro level data of 180,000 people and 70,000 families from a USA national census. The model simulated a yearly change of each individual unit, after which available external data was used in validation before projections were made using the validated model.

Many other MSM have been developed in different countries for various purposes. Some include: TRIM (Transfer income model) [86] in the USA for simulating major government tax and health programs that affect the population, DYNACAN [71] in Canada for simulation private pensions by futuristically projecting their variations, incidence and average level as a function of age gender and birth year. The PBS (Pharmaceutical Benefits Scheme) [103] micro simulation in Australia analyses the resultant government outlays under the pharmaceutical benefit scheme.

These models hold similar structures, i.e, individuals are initialized with micro population data, state change within the simulation is based on transition probability, and individual unit communication and interactions is minimal. Also, though these models

are able to infer from the aggregate output data the implication of tested policies on the individual units, the models are weak in predicting the implication of a consistent or change in behaviour of an individual entity on the whole complex system.

2.2 Agent Based Modelling Technique

ABM is a technique used in modelling systems, comprised of individual, autonomous and interacting entities, i.e, agents. The concept's idea is to model the agents and its environment, such that, the interactions between these entities will generate an emergent behaviour at the macro level, which would be similar to that observed in the target system [110].

Within the last two decades, ABM's have transcended theoretical viewpoints, to more practical applications in various spheres, such as: biology [101], economics [17] and social sciences [14]. Although, ABM considers individual level dynamics, most of the ABMs created before the 2000's[85][22] did not consider using real life data for agent initialization, and agent behaviours. This is seen majorly in their rule creation and inferences, which are analytically assumed rather that being empirically derived from the actual social complex system under consideration [87]. According to Boero et al.[22], this approach raises doubt in the validity of the emergent behaviours produced. Especially, if the intent of the ABM is to create scenarios for testing policies in a social complex system.

According to Boero et al, if a target system is made up of components A, B and C. If component A has features $a_1, a_2, \ldots, a_x$, component B has features $b_1, b_2, \ldots, b_y$ and component C has features $c_1, c_2, \ldots, c_z$, where integers x, y and z are the total numbers of features characterized by component A, B, and C respectively. If the feature com-

bination $a_3 + b_5 + c_7$ produces an emergent phenomenon K_s, closely comparable to a similar phenomenon K_t observed in the target system, then this will imply that, except an empirical data extracted from the target system is used in defining the relevant features of component A, B and C, then it is possible that another feature combination $a_1 + b_5 + c_3$ can produce the same emergence K_s comparable to K_t. Therefore, Boero et al[13] implied that, since an infinite amount of micro specifications (and a consequent infinite amount of possible explanations) within a social complex system can be found to generate a K_s phenomenon comparable to K_t, then empirical data and expert knowledge at the micro level, are invaluable in determining the causal mechanism in play in the phenomenon of interest.

This therefore implies that ABMs that seek complete or partial realism in representing complex systems, need empirical grounding. Yet, very few researchers have developed methodologies to incorporate data in modelling agents[13]. This may be due to the unavailability of behavioural level data[49], or the lack of substantial quality data at micro level[49].

2.3 Micro Level Big Data

With newly established data sources, the world is gradually shifting from a data poor era into the big data era, where individual level behavioural data can be captured in high volume, velocity and variety. Big data analytics is the application of statistical processing and analytics technique to large scale and often real-time data sources for the advancement of understanding real world phenomena, in order to make better judgement[35][3].

This research is interested in the analysis of human characteristic and behavioural

data at individual level, which we refer to as micro level, individual level or granular data in this thesis. Such data is the main input for micro simulations[7] and we intend to investigate the use of such data in creating agent attributes and behavioural rules so as to create agent based micro simulations which are superior in accuracy.

The literature reveals the examples of research that have identified demographics and personality attributes of people from big data, proposing computational algorithms for the cleaning and organization of the data before being used in determining individual level characteristics and attributes.

Mislove et.al.[69] Analysed data on a set of twitter users representing over one percent of the U.S. population. They extracted attributes such as gender, geography and ethnicity from 54 million users with combined historical messages of 1.75 billion. Individual users home towns where deduced using self-reporting location properties within their profiles while the users ethnicity and gender where predicted by mapping names to name-ethnicity and name-gender databases respectively. On the other hand, Chen et. al.[19]. Investigated individual behaviour by predicting personality traits derived from individual historic social media messages. By determining individual level behaviour through individual level data, he was able to create an ad (advert) targeting algorithm which resulted in proving that the derived personality trait deduces from individual message history had the same effect as the personality trait measured by traditional personality questionnaires. Home location of individuals was also predicted by Hamdi Kavak et.al.[45] to a significant resolution by exploring the influence of time span of data collection and location sharing of a twitter user. They collected 92 thousand active user data set containing 77 thousand location foot print, from which a core data set was extracted containing messages sent from user's home location. This data was then cleaned and clustered to extract individual home locations predicted to an

accuracy of 0.87 within a 100 meter resolution.

Individual level behavioural data was also generated by Lovric et.al.[60] Using smart card transaction data collected from a major Dutch public transit authority (PTA). They accumulated a data set containing individual check-in and check-out transactions made at station entrances and exit gates, geographical co-ordinates of stations and transit schedules to determine individual travel behaviours of passengers.

Individual level behavioural patterns have also been generated from mobile phone usage. This kind of data is generally known as call detail record (CDR).

Bassolas et.al.[8] generated human behavioural pattern from big data using CDR. This mobile records where from anonymous users within the metropolitan area of Barcelona. It contained information about the timing of user requests and position of connecting towers. This provided spatial and temporal information of individual users at the point of connection, which was then exploited in extracting individual level travel behavioural patterns.

2.4 Data-Driven Agent Based Micro Simulation Approaches

In this section we delve into the data-driven agent based micro simulation approach to create more realistic and stronger predictive models This approach prioritizes the use of empirical data in all modelling steps, such that in doing this, we take advantage of many of the strategies that have only applied to building micro simulation models in the past, such as modelling entities with micro level behavioural and attribute data in order to mitigate some of the weaknesses of ABM in model validation and policy prediction, while also mitigating some of the weaknesses of micro simulation models by creating simulations that create agent behavioural rules from data. This will enable

a micro-macro relation that produces a model that can predict not just the impact of policies on individuals in a social system, but also predict the impact of individuals on the policy.

In creating a model like this, the significance of empirical data must not be underestimated. Therefore in the next subsection we discuss the data usage approach, as has been highlighted in literature. The objective is to promote the correct usage of data in this hybrid.

2.4.1 Data Usage Characterisation

There are four data usage categories that should characterize a robust data-driven strategy[44]. These are:

Type of data: According to Boero et.al.[13] quantitative and qualitative data types are the two types of data types can be used in data-driven modelling approaches. Quantitative data is characterised by numerical representations, which could be further classified as discrete (e.g such as the number of online followers) and continuous (e.g.$\Pi = 3.14$). Qualitative data refers to qualitatively represented information, such as text and words. A practical example is documented interviews. Datasets such as census Public Use Micro-data Sample (PUMS) files[56][97][53] contain a wealth of quantitative and qualitative data.

Data collection: For more realistic models with the capability for accurate policy analysis, it is important that repetitive data collection be made, as this allows the capture of data change over a temporal span. Hamdi Kavak et.al.[44] Recognizes single snapshot data collection and multiple snapshot data collection, where the former is a one-time data collection and the latter is repeated data collection. The single snapshot usage is

challenging when representing human behaviour, as it does not accommodate changes in behaviour over time, and can therefore only be used in model initialization. Whereas a repeated data collection will allow repeated measurement at different times allowing the capturing of behavioural patterns over time.

Data impact on model: Depending on how the data is used, different changes can be effected on the model. Four types of impact is recorded in literature[87][7][44]:

1. *Initialization*: Where the data does not directly affect the model but is used in initializing agent attributes and agent variables.

2. *Structural modification*: Structural modification is such that the modification on the model is trackable. It could be in form of modifying variables of a formula within the simulation using a machine learning model. For example using a classification model to determine a parameter value.

3. *Structure generation*: The data impact in this case is such that the complete or partial model structure is created from the scratch rendering the change untraceable, for example recreating the behavioural rules using a Bayesian network.

4. *Model validation*: According to [7], model validation using empirical data is such that the result of the simulation correlates directly with empirical data gathered from the target system. therfore such empirical data's impact is in measuring the short comings of the model, which would also aid in model reconfiguration.

The effect of data on the model surges from initialization to structure generation.

Population/Individual level consideration: This is a reference to the level of data

used in creating the agents. This could be either at population level or at individual level. Population level consideration usually involve aggregate level data being used in describing agent properties or actions, such that these deduced properties are distributed the same across the agent population. Individual level agent consideration feeds unique data to every agent in the population creating individualistic agents with unique properties. In social system modelling, population level consideration has been the norm in the past, but with the advances in data collection technology, individual level data has become extractable from target systems, allowing the incorporation of individual attributes and behavioural data in modelling.

2.4.2 Data-Driven Modelling Approaches

The fact that data-driven (DD) approaches to modelling have become common, structured approaches to this technique is still not common. In this section, the volumes reviewed, although, have methodologies applied to particular user cases, their approaches can still be generalized to other problem cases. It was also notice that many of these cases have not prioritized data usage in all their steps.

An earlier methodological effort identified was highlighted in Riccardo et.al.[13]. In which they studied the pre-historical population dynamics and decline of the Anasazi people. This model mostly revealed emergent behaviour that shed light on the social and environmental interactions of these people in order to explain their decline. Agents where modelled from occupants of households in which attributes like age and location where inferred from previous empirical bio-anthropological, agricultural and ethnographic analysis. The entire model was focused on the emergent behaviour on the agent and was not policy focused. Also behavioural data was non-existence and

the agent's behaviours where not empirically deduced.

One of the first attempts to create a structured framework to incorporate big data in ABM was Kennedy et. al.[46], who developed an adaptive simulation architecture which accommodated the use of real time data. Their concept was such that while the simulation is running, real world data is gathered in real time from the target system which is continuously compared with the simulated data for real time consistency, in an attempt to realign and re-calibrate the simulation. In this approach there is no evidence of how the data directly impacts the model, and its focus is on using the data in the validation step alone. The description does not also detail how data can be used in initializing the simulation parameters empirically.

The thesis of Collado et.al.[21] proposed a data-driven agent based approach which focused on empirically grounding the model creation. Their methodology encouraged the data extraction from the real world (case study was Spanish modernization) with the aim of using it in guiding the abstraction process and the initialization of the model. A validation data was also collected separately for validation. Although their approach included the use of quantitative and qualitative data such as interviews, surveys and panels. Their agent consideration was not on the individual level, but rather on the population level and no structural impact on the model by the data was implemented by the simulation.

Smajgl et.al.[90] developed a framework for the parametrisation of human behaviour in ABM, in which he developed 12 distinct sequences for achieving this. Although his work is an improvement on Collado et.al.[21] approach, they provided clear methods within their framework for every modelling step. Even though they accommodated the use of qualitative and quantitative data, they did not discuss repetitive data collection and the data impact on the model only covered initialization and structure

modification while agents considered where not on individual level.

A DD approach was introduced by Singh et.al. [87][15], in which they amalgamated steps used in micro simulation models to mitigate the weakness of ABM in policy making. They aimed at using empirical data in agent based social simulations so as to analyse the transition of family formation affectively. The approach used census data for a particular year to initialize the simulation while using a later census data to validate it. Their approach seems to be strong in using data to initialize and validate the model, but the core behaviour of the agent did not come from historical data.

Bae et. al.[6]created a data-driven approach with a framework focused on using repeated measurements calibrating the model, with the aim of achieving consistent high validation. The model created focused on creating adaptable models by not just tuning parameters but by using the difference between the expected model result and encountered model (created from the updated data) result to choose an agent model within a pre-stored repository, to reconfigure the simulation so as to maintain its accuracy.

Kavak et. al's[44] has been the only author so far that has attempted to create a structured approach for data-driven agent based model, which focused on using individual level data. Although, other authors who have not defined their frameworks have followed in his footsteps. Kavak et. al's[44] created an agent based urban mobility model, using social media data. Although, they had a proof of concept, their approach did not mention a validation process using data, and their model was not validated with real data from the social system.

Data type		Repeat Mea-sure-ment	Data impact on model				Agent consid-eration	Ref.
Qual.	*Quant.*		*Init.*	*Str mod.*	*Str. gen.*	*Valid.*		
No	Yes	No	No	No	No	Yes	Pop.	[13][71][80][72]
Yes	Yes	No	No	No	No	No	Pop.	[46][32][98]
Yes	No	No	Yes	Yes	No	No	Pop.	[21][62]
Yes	Yes	No	Yes	Yes	No	No	Pop.	[90][99]
No	Yes	Yes	Yes	Yes	No	Yes	Pop.	[6][110][39][7]
Yes	No	No	Yes	Yes	No	No	Ind.	[44][87][60][88]
Yes	Yes	No	Yes	Yes	Yes	Yes	Ind.	DDABMS(see chapter 3, 4 and 5)
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Ind.	DDDABMS(see chapter 6)

Table 2.1: Data-driven model approach review

In Table2.1, we compare popular data-driven model methodologies in literature while focusing on their data usage characterisation as discussed in section. It can be seen that almost all the frameworks focus on population level representation of agents. Almost all are open to both quantitative and qualitative data usage. It can also be seen that when it comes to the impact of the data on the model, majority of the approaches focus on using data to initialize their models. Few use data in structure modification (e.g equations), and only [44] has proposed a framework with full data impact on the model. There is still a set back in his approach, which is that, he pays less emphasis on validation with empirical data. To sum it up, the approach discussed in this thesis seeks to focus on creating data-driven frameworks that will use both single and repetitive measurements, allow the full impact of empirical data on the model and create individual agents from individual level data, while also allowing model

validation through empirical data from the target system.

2.5 Chapter summary

This chapter details the state of art research development related to the areas of ABM, MSM, big data and the different research works that have tried to amalgamate some or all of these technologies in simulating social complex systems (SCS).

The first and second sections introduce the concept of micro-simulation and agent based modelling, respectively, including state of art models that have used these individual techniques. The third section talks about individual level (i.e, micro level) big data, while the fourth discusses different solutions that have attempted to structure the application of big data, data mining and machine learning in the amalgamation of agent based modelling and micro-simulations, while characterizing the different data usages in these hybrids.

The approach discussed in this thesis has been differentiated from all other approach as one that focuses on creating data-driven models that will use both single and repetitive measurements, allowing the full impact of empirical data on the model and creating individual agents from individual level data while also allowing the model validation through empirical data from the target system. This will be fully discussed in subsequent chapters.

Chapter 3

A Framework for Data-Driven Agent Based Micro Simulation

In the previous chapter, we have discussed different data driven simulations, their methodologies and short comings, with the aim of highlighting the absence of a methodological approach at the individual agent level to create behavioural rules and initialize attributes from granular/individual level data. Our methodology seeks to fill this gap and also intends to define a simulation approach that adopts the individual level focus of micro simulations and the behavioural interaction focus of agent based simulations to create empirical models that accurately reflects real life social systems after being validated with real life data. All with the intention that such models will be used in policy analysis.

This chapter describes the main element of the proposed framework, using a top-down style view and a conceptual model approach [67] at each level. The first level of the framework gives a high level view of the elements involved in the modelling approach while the lower level view delves into the creation and parametrization of

the agent population using a data driven approach. Finally the individual agent view discussed the processes involved in directly infusing individual level data to initialize the agent attribute and create behavioural rules for each agent, which is ultimately used in simulating the target social system.

In the next two chapters, we give a proof of the effectiveness of this concept by using this framework in the creation of a micro-simulation of the Metro-North Rail road (MNR) using individual level data sets (the published paper can be found in [64]). And also in the creation of a micro-simulation of a distributed computer network (DCN) using individual level user attributes and usage history extracted from the computer network.

3.1 Data-Driven Agent-Based Micro Simulation Framework

3.1.1 High Level View

The figure below (figure3.1) gives an overview of the elements within the proposed framework. It is a conceptual map, highlighting the involved processes in minimal details. These elements are:

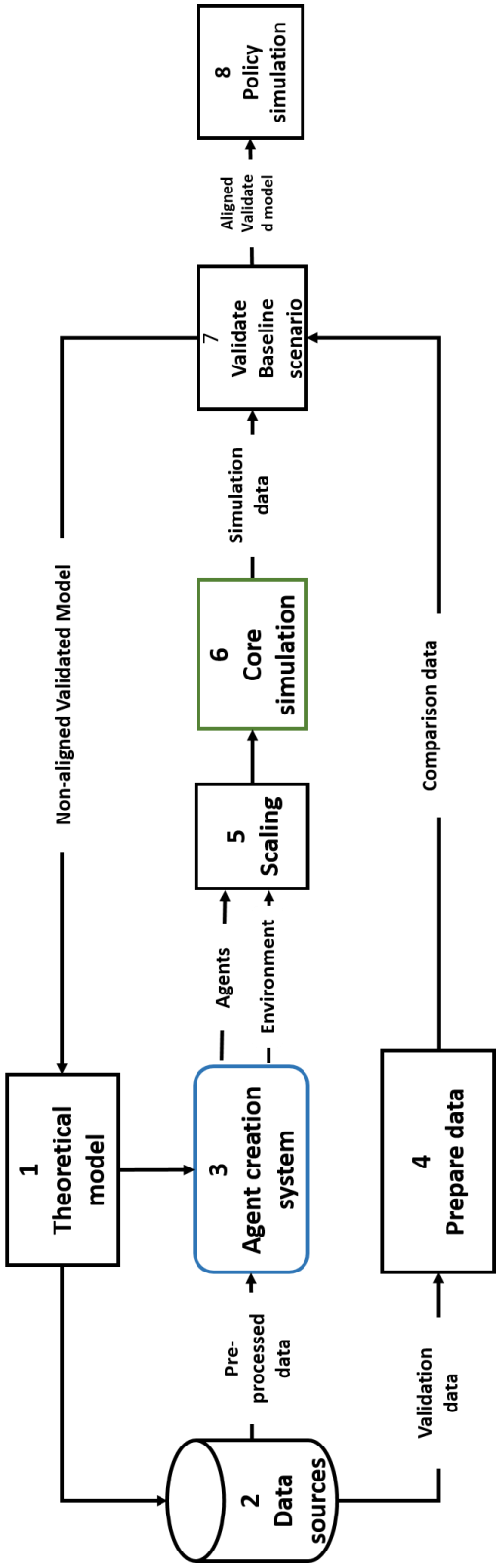

Figure 3.1: Conceptual framework of the proposed data driven micro simulation approach, from a high level view

The theoretical model: This is an an experimental design for the systemic exploration of the theoretical issue of interest within the social complex system being targeted. It describes in detail the context of the target system with a focus on the purpose and goal of the simulation. It gives a skeletal window describing every entity within the simulation including their action, attributes, behaviours and purposes, as described in [67][34]. The theoretical model is key in determining and selecting suitable data sources for the initialization of agent attributes and the generation of agent behavioural rules. Also, in a case where the output of an already simulated baseline scenario does not align with the validation data, it will mean the simulation is inaccurate. In such a case, the model is cross checked against the theoretical model, so as to determine the fundamental cause of error. The theoretical model determines the fundamental assumptions followed by the model, and drives the data collection strategy of the whole simulation project. In simple words, the theoretical model is the architectural representation of the whole simulation project.

The data source: This is a system or repository that generates or contains the individual level data needed in creating the agents. For the required agent behaviour and attributes to be generated, this data must be able to provide granular level information of the real world entity being modelled. This allows the discovery of behavioural patterns for agent behavioural rule generation and attribute initialization [33]. A data source could be as complex as a physical devise, gadget, APIs or software generating data or as simple as a repository containing downloadable files. The data collection plan must also include a data source for collecting validation data alongside, which must have the capability of checking the validity of the baseline emergent behaviour, generated from the overall simulation, before simulating other scenarios for policy analysis. This data is cleaned and transformed into a format that enables easy comparison with the

simulation's result.

The agent creation system: This uses the theoretical model and the raw data from the data source to create the agent population and initialize the environment, with each agent reflecting the real life entity in the target system. This is explained in details in section3.1.2

The scaling process: This process scales the agent population appropriately, in ratio to the actual target system being represented. Due to limitations in data collection, it is difficult to get enough data to represent every element in the complex system. Therefore the scaling process attempts to scale the model in the right ratio. This ratio is achieved by comparing the available data, to the actual size of the target system. Take for example, makinde et.al[64], while simulating a passenger rail system of about 100,000 passengers, they where only able to successfully extract the travel information data of about 12,000 passengers. Therefore, the whole model was scaled to 1:10, so that each passenger agent represented 10 real life passengers. The simulated environment of the rail system (such as the number of seats in a train carriage) was also scaled down by 10% as appropriate, so as to ensure an accurate reflection of the system.

The core simulation process: This process simulates the baseline scenario, so as to produce an emergent behaviour that will reflect the target systems behaviour. The data from this simulation is then compared to the validation data set in the validation process stage, before other policy based scenarios are simulated.

Validate baseline scenario: The objective of this process is to validate the baseline scenario model, by measuring its accuracy. The baseline scenario represents the the social system as captured by the data. This base line scenario, is simulated in the core simulation process (process 6), and is validated in this process. To do this validation, the output data, taking from the core simulation, is compared with a validation data

set, extracted from the target system. Depending on the nature of the data set, different comparison techniques can be used, such as root mean square error[107] and R2 coefficient of determination[74] e.t.c. The aim to meet an accuracy threshold, which if not met, will mean the simulation does not align with the target system. In such a case, The theoretical model will be investigated, and the simulation rebuilt. **policy simulation**: In this process, alternate scenarios, different from the baseline scenario, are explored. The process involves changing model parameters, so as to create 'what-if' scenarios that cannot be tested in real life. From this process, different social policies such as price changes e.t.c. can be tested.

3.1.2 Agent Population Generation View

Figure3.2gives a second level detail of the agent creation system (node 3.0 in Figure3.1), highlighting the data flow between interrelated elements that produces the agent population. The agent creation system entails 4 main processes, which include: acquiring and preparing data (process 3.1), attribute fitting/model training (process 3.3), environment attribute initialization (process 3.6) and modelling individual agents (process 3.4).

In the data acquisition and preparation process, the raw data (pre-processed data) from the data source(s) needed for the simulation is extracted. This would include the data needed to create the identified agent and initialize their environment attributes. Depending on the data source, this process might be as straight forward as downloading files from an on-line repository, or as complex as streaming data from authenticated websites. The primary goal in this process is to clean the extracted data by handling

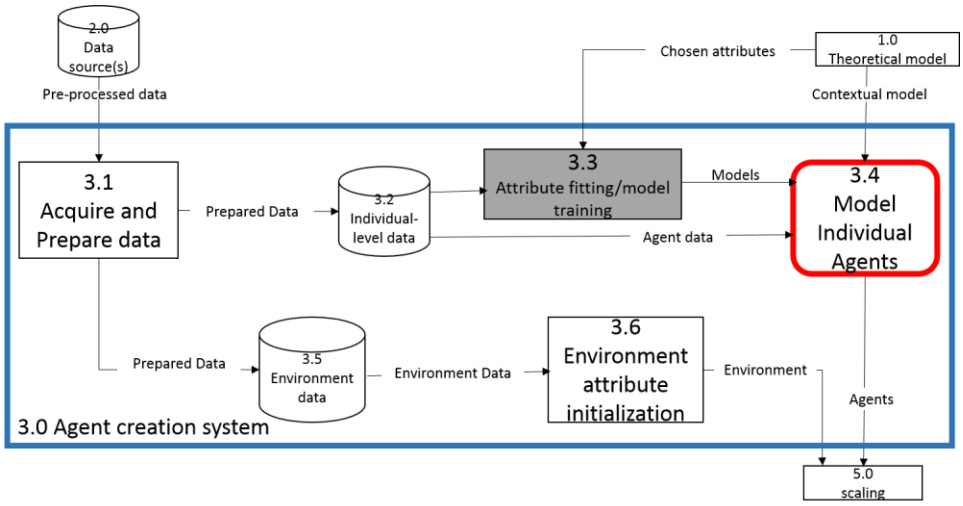

Figure 3.2: Conceptual framework of the proposed data driven micro simulation approach, from an agent population level view.

missing values, removing unusable and noisy entries and transforming data fields. The output of the process would be the individual level data for agent creation and the corresponding environment data which could then be stored in temporary repositories to avoid data loss.

The attribute fitting or model training process is an optional process. It is the data mining task of training a machine learning model to infer the value of a chosen attribute. The process would require gathering a training sample of the targeted attribute from outside sources, this is then trained to create a model for predicting the attribute initialization value.

The environment attribute initialization process uses the processed environment data to initialize the agent environment.

3.1.3 Individual Agent Creation View

The conceptual model for the creation of the individual agents using the individual level data can be seen in figure3.3.

Using the theoretical model from which the simulation idea is drawn, process 3.4.1 (classify agent type process) divides the prepared agent data into the various types necessary for creating the different types of agents involved in the simulation. The data is then separated in the next process into data needed for attribute initialization and that needed for behavioural pattern recognition. The attribute initialization process (i.e. process 3.4.3) first transforms the individual level attribute data into a format suitable to be used in any attribute initialization technique. Three major agent attribute initialization techniques have been commonly used in literature [7][45][33][51]:

1. **Model based initialization**: uses machine learning and statistical models to infer the chosen attribute value. Such technique was used in [33] who sought to simulate user behaviours in on-line social networks. The user-agent attribute value was predicted using a sentiment analysis model, which predicted the sentiment value of each agent's micro blog.

2. **Look-up based initialization**: uses a look up table to determine the chosen attribute value, as used in [45], where a name to gender look-up table was used to determine the gender of a twitter user from the user name.

3. **Direct initialization**: is a direct technique, where the value of the processed data are simply passed as the agent attribute value. This technique was used in [7] where the agent's initial attribute value for sex and age where directly extracted from the South Korean population census data set.

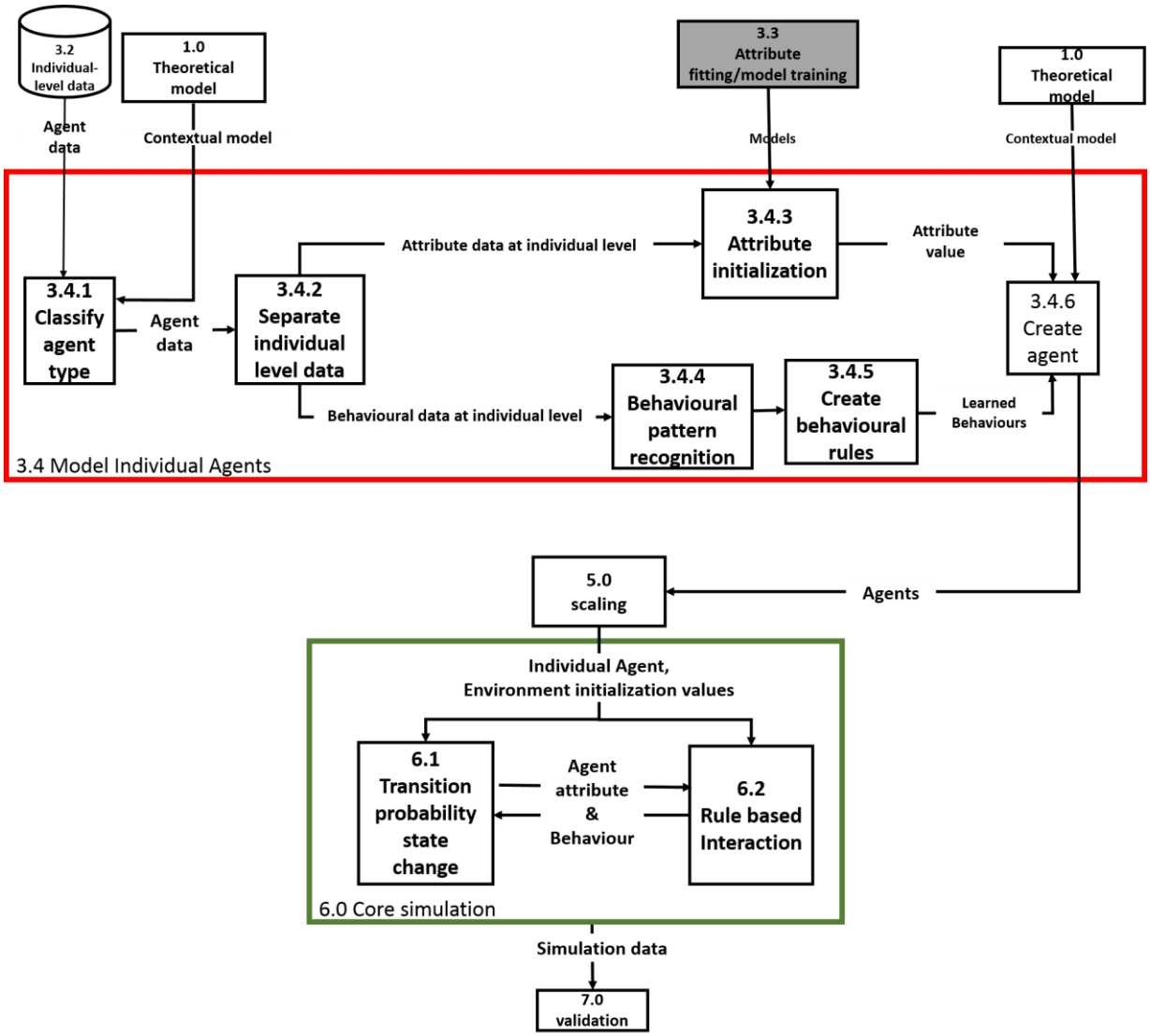

Figure 3.3: Conceptual framework of the proposed data driven micro simulation approach, from an individual agent level view.

The behavioural level data, which is the individual level data containing information from which the actions of the target entity within the target system can be extracted, is used by the behavioural pattern recognition process 3.4.4 to learn individual level behavioural patterns. This process could be achieved using machine learning

models or statistical inferences. These discovered patterns are then encapsulated into programming language statements as behaviour functions.

The combination of the learned behaviour rules and attribute values are then used in modelling the actual agent before scaling the agent population, after which the baseline scenario is then simulated.

The process 6.0 (core simulation process) combines both the transition probability state change of the micro simulation technique and the rule based interaction of agent based model to implement the core simulation process. In both micro simulations and agent based models, their inputs are affected by the features of their environment. In both cases, outputs consist of responses to a combination of inputs and their current states. They both have operating characteristics for controlling the transformation of inputs to output. For ABMs, the operating characteristics are typically implemented in a set of rules while for micro simulations, the input to output transformations are usually accomplished by transition probabilities. The simulation processes 6.0 and 8.0 (see figure3.1) in our approach embrace these two operating characteristics so as to create a more stochastic and data driven approach that will produce more realistic models that reflect targeted SCS.

3.2 Chapter Summary

After subsequent chapters have revealed that the combination of the micro-simulation technique and the agent based approach can create stronger hybrid model that amalgamates ABM and micro simulation techniques in creating models that have a micro-macro bridge such that they are strong in predicting how the policy affects the agent and how the agent behaviour affects the policy as a whole, we therefore in this chap-

ter have introduced and discussed a novel methodology that will amalgamate both the ABM and the micro-simulation technique using a data-driven concept that focuses on the accurate representation of every individual entity in the target system. The aim was to fill in the gap of an absence of a methodological approach at the individual agent level in creating behavioural rules and initialize attributes from granular/individual level data.

The next section is a case study that uses the described framework to create a data-driven micro simulation of a passenger rail system. This work has been published as a chapter in the Springer book series: Advances in Intelligent Systems and Computing [34].

Chapter 4

Simulating a Passenger Rail System Using the Data-Driven Micro-simulation Framework

In this chapter we create a traffic demand simulation using the framework discussed in the previous chapter. The aim of the simulation is to create a base model that predicts the travel demand of the target public transport system. This base model is then used as a test-bed to simulate and analyse the travel demand impact of seven common pricing policies used by public transport organisations. We use the activity based approaches[68] to define the agent (passenger) travel behaviour, which is an approach drawn from a belief that a passenger's travel choices are rooted in his deep intention to finish an activity. From this philosophy, we believe his behaviours will emerge.

The simulation is built using the MATSim (Multi Agent Transportation Simulation) Java library, developed to provide a structure for implementing large-scale agent-based transport simulations. Currently, MATSim offers a framework for demand-modelling,

agent-based mobility-simulation (traffic flow simulation), re-planning, a controller to iteratively run simulations as well as methods to visualize some outputs generated by the modules. A thorough description of MATSim can be found in Horni et. al [37].

4.1 Motivation

Due to consistent increase in urbanization and societal changes in economic status, tremendous pressure has been mounted on transportation infrastructures, leading to a rise in research on travel demand models (i.e, simulation models created to reflect the travel behaviour of a population), as opposed to continuously building new infrastructure to meet growing demands. The main challenge in building travel demand models is to capture passenger travel behavioural patterns. Different approaches have been used over the years to achieve this.

The first generation of travel demand models were popularly called four step models, developed in the 1960's [81]. These models had obvious limitations, in capturing travel behaviour, making them obsolete within a short period of time. Firstly, they used aggregate level data to infer traveller's behaviours. They also considered trips separately, ignoring the fact that an individual's travel behaviour is based on a wholesome plan which considers every trip within his entire day journey. Also, their step sequences had inconsistent sub-modules, and in the heart of the models lay aggregate procedures [50], which altogether limited these models in policy impact prediction. This led to the development of two other approaches, i.e, dis-aggregate choice and the activity based approach for capturing travel behavioural patterns of travellers in travel demand models [40]. Although dis-aggregate choice micro-simulation models sought to use individual level data and were accepted for a while, but they maintained

the fundamental error of their predecessor, analysing each trip independent of another trip made by the same individual, thereby not being able to capture the full traveller behaviour pattern of each individual traveller, and thereby compromising the model's capability for policy testing.

The activity based approach on the other hand sees the traveller's behaviour as a derivative of a need to be involved in activities distributed in time and space[81]. It accommodates the use of individual level data to extract travel behaviours of each individual traveller so as to create empirical simulations that accurately reflect the target transportation system. By using the activity based approach to travel demand modelling, we will be able to combine the advantages of micro simulations (policy focus, transition probabilities, individual level data usage) with those of agent based models (emergence focus, environment interactions, rule based) to create an individual level empirical hybrid that realistically determines the effect of policies on the individual agent, and the effect of individual agent behaviours on the policy and complex system as a whole.

4.2 Aim and Objective of Simulation

We intend to evaluate the use of the framework discussed in chapter 3 in creating a hybrid model that combines ABM and micro simulation techniques in creating models that have a micro-macro bridge such that they are strong in predicting how the policy affects the agent and how the agent behaviour affects the policy as a whole.

To do this, we intend to simulate the transportation system of a real public transport company in New York USA (see figure4.1) called the Metro North Rail road (MNR), operated by a company called Metropolitan Transportation Authority (MTA). We use

the real life passenger survey data set of the stated company, in which each passengers has recorded his spatial-temporal information of every activity travelled to or coming from within a space of 3 months. This is then used as our individual level data set used in modelling the unique travel behaviour of each passenger, thereby creating an agent population that truly reflects the demand on MNR's system. The attributes of the agent environment, consisting of the real life coordinates of the train stop stations, rail track links, route time tables and the attributes of every individual train (each train's speed, capacity, engine size and schedule) within the system are all initialized from empirical data. All these are combined to create a game-like interplay between demand and supply that will reflect the actual behaviour of the target complex system.

We intend to use the framework to create a simulation that takes advantage of the individual level characteristics of the micro simulation technique and its stochastic process for state change to simulate micro level passenger agents that make realistic travel decisions. We also intend to use some of the rule-based procedures borrowed from agent based modelling technique to allow agents interact with themselves and the environment. The combination of this will create an overall emergence that can be used in testing policies whose impact can both be estimated on the micro (individual agent level) and macro (global policy level) level.

Figure 4.1: Metro North Rail Road New York

In the subsequent subsections we breakdown the data-driven agent based micro-simulation creation process using the novel methodology created in chapter1

4.3 Simulation Methodology

4.3.1 Theoretical model

Following process 1.0 in chapter 3, each passenger of the rail system is modelled as an individual agent, travelling from one activity (e.g, “Home”) to another (e.g, “Work”).

Three types of activities have been identified from the data, which will allowed a further classification of the passenger agent, based on their activity goals:
Home-work-Home: passengers travelling to work and back home
Home-School-Home: passengers travelling to school and back home
Home-Leisure-Home: passengers travelling to leisure and back home

The trains moves from train station to train station using the real live time table of metro northern rail road (MNR), stopping at stations to pick up passenger agents. An early arrival vehicle will wait till the next departure time before leaving the station. The x-y coordinates of the location of train stations and passenger activities are gotten from a look-up table that will be further discussed in subsequent sections. A pictorial view of the movement of passenger agents can be seen in figure4.2, while skeletal model of the agent can be seen in figure4.3.
There are three important pieces of this simulation:

1. Each passenger agent generates his initial day plan, encoded with information of his activities (i.e, duration, route and activity end times within a day), gotten from the travel diary survey of the corresponding real life passenger. An example of an initial day plan for 20 agents can be found in[63].

2. All passenger agents execute a plan simultaneously in a virtual model of the physical rail system. Therefore agents can miss the train, be late for an activity or not be able to get in the vehicle if it is already filled to capacity.

3. Agents learn from experience by performing an iterative process on their day plans. An iteration is completed when the agent travels through all the activities listed in his day plan. The scoring of the plans is described in detail in subsection

4.3.8. So for every iteration the agent could either chose any of the old plans using a discrete choice model or create new plans by modifying existing ones (modification can be done by changing time or route but not activities). The iterative process is repeated until the average population score stabilizes.

A basic assumption of this simulation is the assumption that customers have no alternative mode aside from this public transport train line. Figure4.2below is a visualization of how an agent can move within the simulation

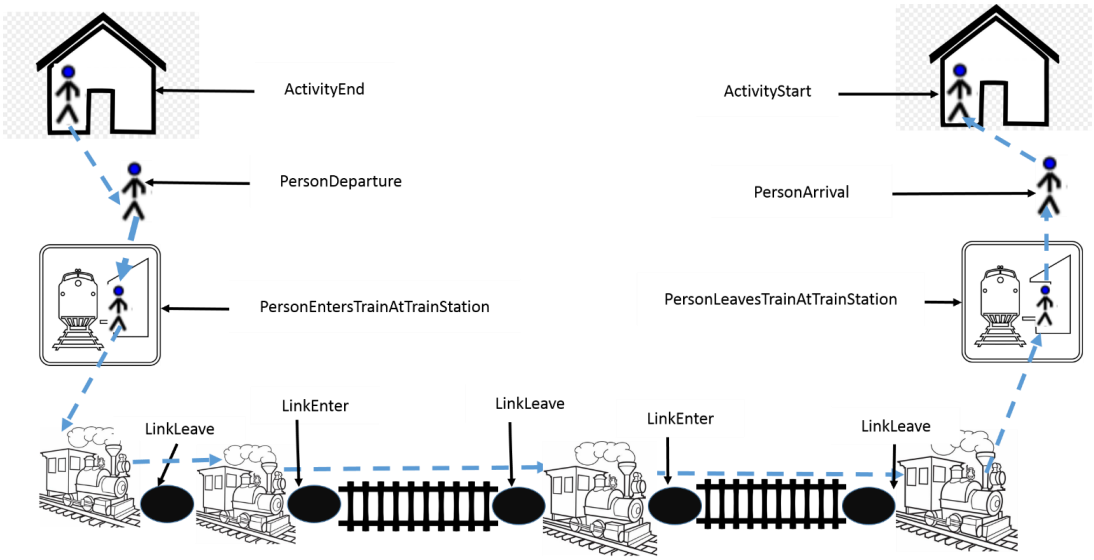

Figure 4.2: A pictorial view of the movement of the train, and passenger agent. Shows how an agent moves from his first activity after it has ended, to his second activity, using the rail system. This process constitutes different events including arriving at the station, entering the required train, passing through rail links while in the train, arriving at the destination activity rail station and arriving finally at the destination activity

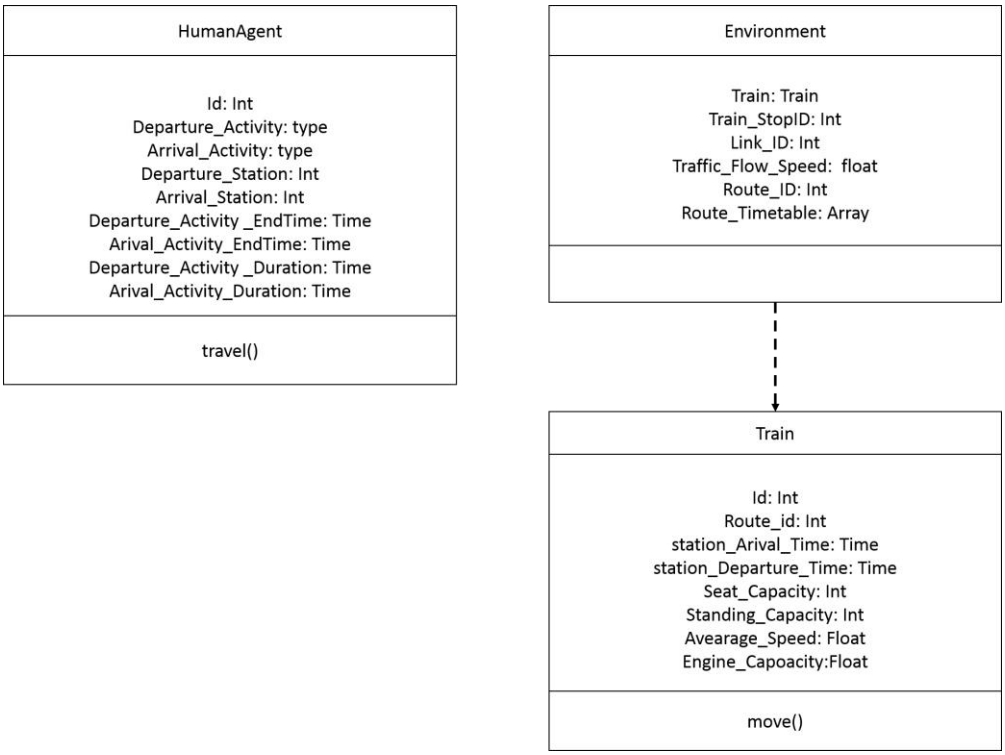

Figure 4.3: A simple theoretic model of the simulation. It shows the attributes of the human/passenger agent, train agent and the environment

4.3.2 Data source

Following the framework as represented in subsection3.1in chapter 3, we discuss how we followed process 2.0 in establishing our data sources. To create the day plan for each agent, we use a travel survey data set made openly available by Metro-North Commuter Rail road (MNR)[82]. This survey was conducted on board every inbound train operated on the Harlem, Hudson and new Haven routes in 2007, and customers filled forms detailing their trip boarding times, alighting times, travelling purpose, and how often they use the rail system. The survey data set contains 92,726 rows and 99 columns, all stored in a Microsoft access database. Each column represents the survey questions asked during the passenger survey, which was executed during train service

hours. Each row represents each passenger's data, who participated in the survey. After pre-processing, we were able to capture each passenger's travel purpose, station check-in, station check-out and activity duration and purpose. The station's geographical location and the rail network map was extracted from OpenStreetMap, while train schedules and capacities were gotten from the public transport authority's(PTO's) website. In addition to the surveys distributed, MNR field workers created a count data set from counting the total number of passengers getting on, and the number total number of passenger's getting off each train, and at each stop. Our simulation was validated by comparing passenger agent's arrival and departure volume, to the real life passenger count data. The validated baseline model was then used in predicting passenger behaviour when different pricing strategies were applied.

4.3.3 Agent creation system

This subsection, further discusses how the agent creation system (i.e. process 3.0) was used in creating an agent population, with each agent reflecting each unique passenger that then cumulatively represented the passenger demand on the rail system.

Acquire and Prepare data Following Process 3.1 in figure3.2, the data set acquired for this simulation were static files download on-line. These include:

- *Environment data-sets*: an XML representation of the rail network of Metro Northern Commuter Rail, describing the physical layout of the network and the coordinates of its stations and links between them was downloaded from OpenStreatMap [79]. This file was cleaned to remove road networks, tramps and every other element excluding the MNR rail networks and stations. The x-y co-

ordinates of the train stations where extracted to serve as a lookup table for the train stations used by individual agents

- *Train data-sets*:Two General Transit Feed Specification(GTFS) files [63][82] containing the transit system's latest daily schedule and physical characteristics of all trains (individual train capacity, number of seats per carriage, standing area capacity and route) vehicles on chosen active transit lines were also downloaded.

- *Human(Passenger) agent data-sets*:The survey data-set contains 92,726 rows (every passenger that filled the survey) and 99 columns (each column is a question represented in the original survey filled by all passengers), all stored in a Microsoft access database. This was downloaded as open access, from the MNR website [82]. The columns where filtered to leave only those useful for this simulation, such as: Begstationidr, endstationidr, ob1stationidr and ob2stationidr columns, which contain information on the passenger's frequent alighting or departing station, depmam-r and outdepmam-r columns, which has information of the passenger's activity scheduled start time, and the depmam which is the train's boarding time.

4.3.4 Model Individual Agents

Following figure3.2, process 3.4, at the individual level, we use the passenger travel survey data-set to extract an O-D matrix (origin destination matrix) of each passenger activity, to create an overall agent plan for the day, which will be a summary of the agent's basic travel behaviour, centred on the passenger's day activities. This will serve as the initial demand that will be passed into the core simulation module.

Classify Agent Following process 3.4.1 in figure3.3, the traffic demand on the rail system is generated based on the concept of daily activity demand from which the need for transportation is derived. Such activity could be activities that require you to be at home (Home activity), work (Work activity), or leisure(Leisure activity) etc. We segmented customers based on their travel patterns:

- *Trip based passengers*:Customers that were only taking a single journey but were not coming back through the train system.
- *Tour-based passengers*: Customers that were departing and arriving using the same stations, and were either leaving from home or would finally arrive at home at the end of the day.
- *Pattern-based passengers*:These are tour-based passengers that use the system at least twice a week.

Attribute initialization Following process 3.4.3 in section3.3, we get the agent attribute initialization values from the data. The attributes are represented as a $tuple l, b, e, \Delta b, \Delta e$. The closest station to the activity location is represented as l in x-y coordinates. This could either be extracted from the Begstationidr, endstationidr, ob1stationidr or ob2stationidr columns in the survey data-set. The station name is first extracted using the direct initialization method after which the look up method is used to get the x-y coordinate of the stations. The scheduled start time of the activity is represented as b, extracted from the depmam-r column of the survey data-set. The scheduled end time is represented as e, extracted from outdepmam-r column. Train boarding time at the beginning of the activity is represented as Δb, extracted from depmam column. The actual duration $t_{dur,q}$ of preferred activity q is b–e (the difference of e and b).

It should be noted that an activity is assumed to start at the time the passenger boards the train. (NB during the simulation process, the start time and the route to an activity can be altered, but the duration cannot. Therefore, plans with activities whose duration ends up surpassing its preferred end time due to chosen routes or start times will be scored as bad plans and would be likely discarded as superior plans are generated).

4.3.5 Behavioural pattern recognition

Following Process 3.4.4 in Figure3.3, to extract behavioural patterns of agents, different strategies can be used including machine learning and statistical inferences as explained in the last section. In this simulation, the behaviour of the passenger agent is reflected by his pattern of journey purpose and mode of transport. Our data captures the journey purpose and the mode. A major assumption is that the activities locations are at the train stations stated by the passengers in the surveys. In most cases, simulations like this will use a probabilistic models to define the agent's activity choice, but because of the simplicity of our data, in that the passengers have noted the activity frequently involved in while using the MNR's rail system, we simple use this activity with highest frequency coupled with the activity the passenger always starts from and ends up with. We consider this activity the "Home" activity.

For our simulation we decided to use only pattern based passengers in creating our agent population, as this is the group that places the most consistent demand on the rail system, and the group from which plans can also be generated that start and end at home (this is a crucial behaviour in activity based demand generation). This will give frequent journey patterns of: $home....outing\ home$. Where outing represents

the major journey purposes extracted from the customer survey data, which are work, leisure/recreation, school, personal business and other. For our simulation we only used passengers going to/from “work”, “leisure” and “school”, in an aim to extract more consistent passengers.

4.3.6 Create agent

Using the attribute value and behavioural rules of each agent, MATSim Java library [37] is used in generating each plan file. A plan file is an XML file attached to each agent, to direct the agent’s travel behaviour. It contains the basic geographical and temporal movement of every agent, and also its mode of transportation per time. From Figure4.4, the agent called- person 1254 -starts his travel from a home location at coo-ordinate x,y (435426, 560687), at time 06:40am. He takes a public rail transport leg mode pt to a work location at coordinate x,y (335426, 511187), after which he then finishes this work activity at time 7:40pm and takes the train back to his home location. The agents route, departure and arrival time is was is continually optimized in the core simulation, in process 6.0 of figure 3.3 in chapter 3. We assume the agent’s activity takes place at the final station, from which the agent alights to indulge in the activity. The combination of the agent plan, of all the agents in the simulation, is revered to as the demand on the rail system.

```
<plans>
 <person id="1254">
   <plan>
    <act type="home" x="435426" y="560687"
      end_time="06:40:23" />
        <leg mode="pt" />
    <act type="work" x="335426" y="511187"
      end_time="19:40:38" />
        <leg mode="pt" />
     <act type="home" x="435426" y="560687"
                />
   </plan>
  </person>
</plans>
```

Figure 4.4: This is the XML format of the initial agent plan produced when the passenger agent is created in process 3.4.6. The mode of transportation referred to as "pt" is short for public transport, which is the rail system being simulated.

4.3.7 Scaling

In Process 5.0 as shown in figure3.1, scaling is applied. Due to the fact that after pre-processing the passenger survey data set, only 12,000 records where fully preserved, therefore, only this amount of passenger agent models could be created. Since this number is close to 10% of the whole data-set (the whole data-set is assumed to be the average daily volume of the rail system as recorded in [82]), we then scaled the system supply or environment in response. This is achieved by reducing the number of seats on each train to 10% of its capacity, in order to match the agents represented. Using this process, one agent will end up representing 10 real life passengers. This has been

proven experimentally by [37], not to have a substantial effect of the simulation.

4.3.8 Core simulation

Every process up until the core simulation process has helped to create the model of the MNR passenger rail system by modelling the system's travel demand (agent population and their individual initial travel plan) and network supply (the agent's environment, including the trains, stops, tracks, etc.). The core simulation process is where the actual simulation of the demand and supply model is done. The simulation uses both the rule base technique of agent based simulations and the probabilistic state change of micro-simulation, such that the whole simulation works as a Monte Carlo engine.

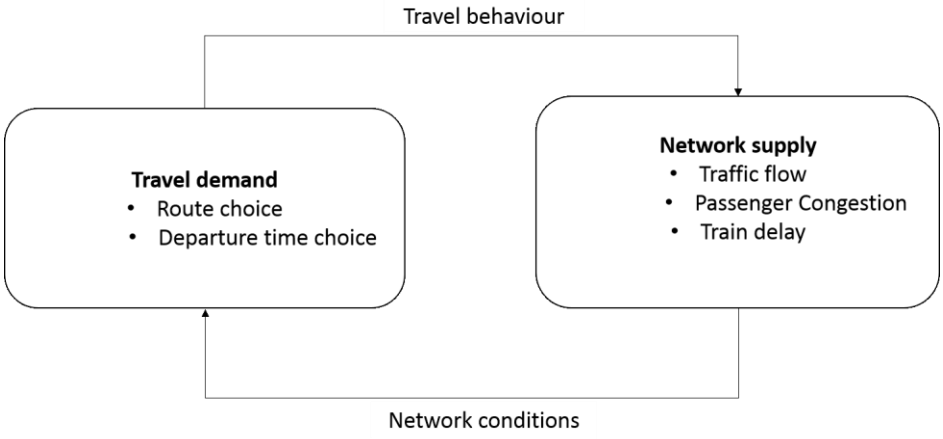

Figure 4.5: Demand supply perspective of the MNR passenger rail simulation

The simulation process is an iterative process, where the passengers keep adjusting their plans to optimise their usage of the network system and reduce activity arrival lateness. This iteration will go on till an equilibrium is reached where no better plans can be created or chosen from the list of already created plans. In other words, after every iteration, agents adjust their behaviour for their own benefit and only stop doing

that when further improvement is minimal. This tends to mimic the actual long term decision making of consistent travellers who will make decisions that will eventually optimise their travel plans so as to get to their activities early enough.

Scoring an agent's day plan An agent's day plan is scored after an iteration, in an attempt to weed out bad plans so that only viable day plans evolve to further iteration steps in the simulation [37]. The value of score S of an agent's plan is represented in equation4.1

$$S = \sum_{q=0}^{N-1} (s_{act,q} + s_{act,q}^{late}) + \sum_{q=1}^{N-1} S_{trav,Rail-PT(q)} + U_{mon} \tag{4.1}$$

Where: N is number of activities, which is 2 (e.g. home and work or home and school), $s_{act,q}$ is the utility for performing activity q, and $S_{trav,Rail-PT(q)}$ is the (dis)utility (i.e. negative utility or penalty) for travelling to activity q from the location of an activity $q-1$ using the rail public transport mode. This implies that routes with longer distance will be less attractive as this will reduce the overall score. U_{mon} Is the (dis)utility based on the total monetary cost of the combined day trip (therefore higher transport cost will reduce the overall cost of the plan). The penalty for performing activity q late is given as:

$$S_{act,q}^{late} = \beta_{late} * t_{late} \tag{4.2}$$

Where t_{late} is the duration of lateness and β_{late} is a negative slope (we use the MATSim default of $-18\$\backslash hr$). The utility earned from performing an activity is given by:

$$S_{act,q} = \beta_{dur} \cdot t_{typ,q} \cdot \ln\left(\frac{t_{dur,q}}{t_{dur,q}}\right) \tag{4.3}$$

Where: $t_{dur,q}$ and $t_{typ,q}$ are the actual and typical duration of activity q respectively. The actual duration for activity q was extracted from each passenger's survey and mapped to the corresponding agent while the typical duration was the highest occurring duration of the activity q among the surveyed passengers. β_{dur} Is the marginal utility of the activity duration or time as a resource (we use the MATSim default of $6\$\backslash hr$). $t_{0,q}$ Is the minimal duration. The mode-specific utility from travelling is described by equation4.4

$$S_{trav,mode(q)} = C_{mode(q)} + \beta_{trav,mode(q)} * t_{trav,q} \tag{4.4}$$

Where: $C_{mode(q)}$ is the alternative specific constant represented as 1, $t_{trav.q}$ is the travel time, $\beta_{trav,mode(q)}$ is the direct marginal utility of time spent travelling. This value is 0 since we are only concerned about the rail mode.

After the plan has been scored for each agent, it is then stored in the agent's memory. Plans selected for the next iteration are either previous plans whose scores have been improved either by changing the agent's route or adjusting its activity departure time, or plans picked from the agent memory using a multinomial logit (MNL) model. The proof for the equations stated above is beyond the scope of this work. It can be found in Horni et al. [37]

4.3.9 Rule based interaction

Following process 6.2 in Figure3.2, this module is where the mobility simulation is executed such that virtual transit vehicles are simulated according to their schedules. A queue model is used to simulate the traffic flow such that when the agents enter into the train the train travels through a link (i.e. the rail) in free flow speed and only comes out

(gets to its next station) at the time required to travel through the link. Therefore agent link transition is possible if capacity of the link and the train allows for agent to enter it. Agents board these train vehicles in order to get to their activities within their daily plans. When this is over, the agent's daily plans are scored (explained in previous subsection) by incorporating people's preference such as price sensitivity, route and late/early arrival to an activity. Passenger agent interaction within this environment can lead to consequences such as lateness. Also, a passenger will not be able to enter a train if it is full or if it is not going to get to his activity train stop. He will only board his train on or before his departure time, and his departure time could be adjusted in the creation of a future plan in the event that the departure time amounts to the passenger being late for his activity. To avoid this, agents may modify copies of their plans after each simulation iteration, in order to avoid situations that lead to bad scores. Such changes could include: change in departure time or change in route. Not all plan copies are discarded after each run, each agent has a maximum of 5 plans within the working memory.

4.3.10 Transition probability state change

In this simulation, the state change we are refereeing to is the probabilistic plan change that happens after every iteration, such that a new plan is picked from the unique plans of the individual agent stored in memory. This choice is made using a multinomial logit Model.

$$P(i \mid C_n) = \frac{e^{\mu S_{ni}}}{\sum_{j \in C_n} e^{\mu S_{nj}}} \tag{4.5}$$

With μ always controlling the preference for higher scores. Where i is a plan id,

for agent n, which is a part of the set of plans C_n within the working memory, j is the size of set C_n, while S_i is the score of plan i. **Running the simulation** We ran our experiment on a desktop PC with an Intel i7 3.6GHz processor and 8 GB RAM. We ran 200 iterations to bring the simulation to equilibrium. The simulation was scaled down to 10 percent of the original passenger daily volume for MNR (this was because, after pre-processing the survey data, only about 12,000 records were usable). In response to this, the physical system model was also scaled down so that one agent will represent 10 individuals within the system. This will have minimal impact on the simulation results [37]. Figure4.7gives a graphical view of the simulation as it approaches equilibrium, as all the agents optimizes their plans in each iteration.

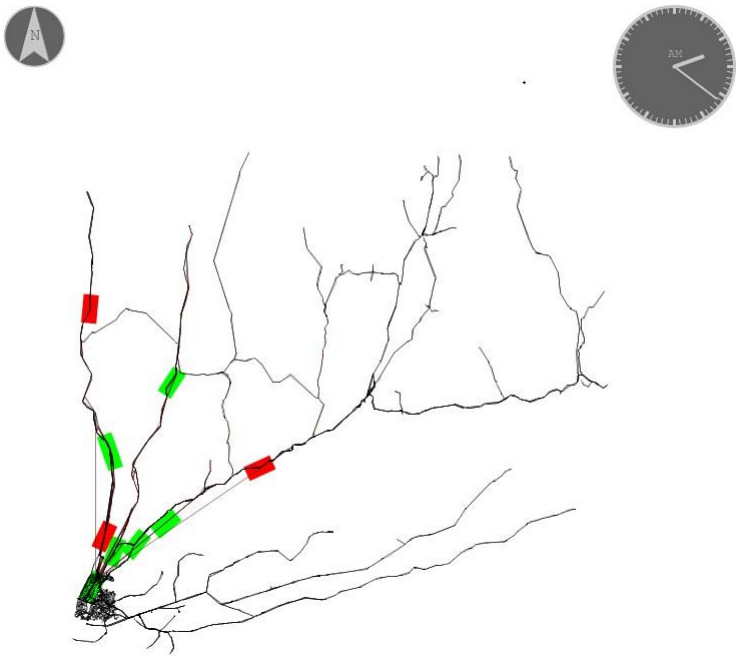

Figure 4.6: A visualization of the running simulation. The web signifies the rail tracks while the coloured boxes is a representation of the moving trains the actual passenger agents are inside the train. It should be noted that the colours are insignificant, and are an inevitable product of the visualization software used.

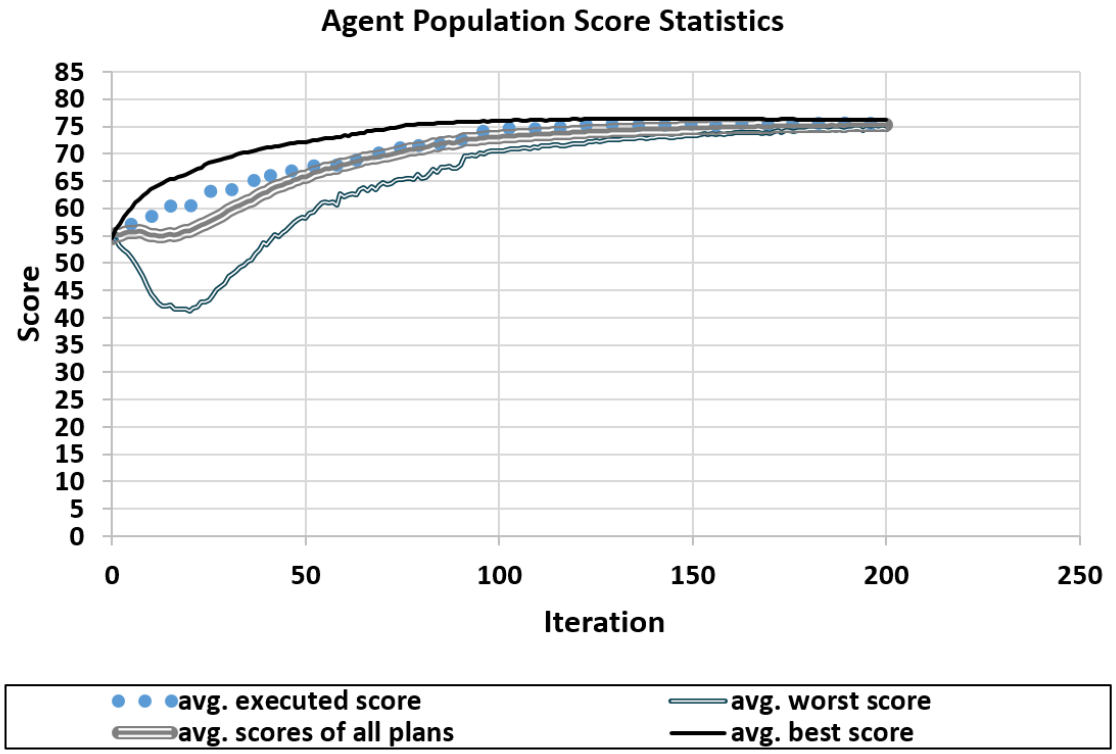

Figure 4.7: The score statistics gives a graphical view of the scores of the agent population daily plans. This gives a picture of the evolution of the plans, as they get better and better till they reach an equilibrium phase where each agent seems to have reached its best plan, which cannot be improved.

4.3.11 Validation

In addition to the surveys distributed, MNR field workers created a count data-set from counting the total number of passengers getting on, and the number of passengers getting off. We compared the simulated volume percentages of agent's arriving and departing at peak hours bins to that of real passengers getting off and on the trains. The bar chart for this can be seen in Fig4.8and4.10. Also, in an attempt to measure the accuracy of the model we analysed the correlation of the simulated data to the count data in4.9and4.11. Both arrival and departure data from the simulation seem to have high associations with the count data equivalents, giving correlation co-efficiencies of 0.88 and 0.95, respectively. The simulated data is not expected to correspond perfectly with the count data-set, due to the fact that not all the counted passengers volunteered to fill the survey form. But it can be deduced that the volume of simulated agents getting off the train (simulated arrival), peak in the evening at about 6pm, while the volume of simulated agents getting on the train (simulated departure), peak in the morning at about 8am. This corresponds to the real passenger count conducted.

The highest level of difference can be seen in the evening peaks of figure4.8(especially at 5pm). There seems to be a sharp arrival change between the 5pm bin and the 6pm bin in the passenger count dataset. This is not reflected in the simulation results as the simulation predicted such sharp change an hour too early i.e. between 4pm and 5pm.

In the next section, we use this model to analyse the impact of different pricing policies on customer behaviour.

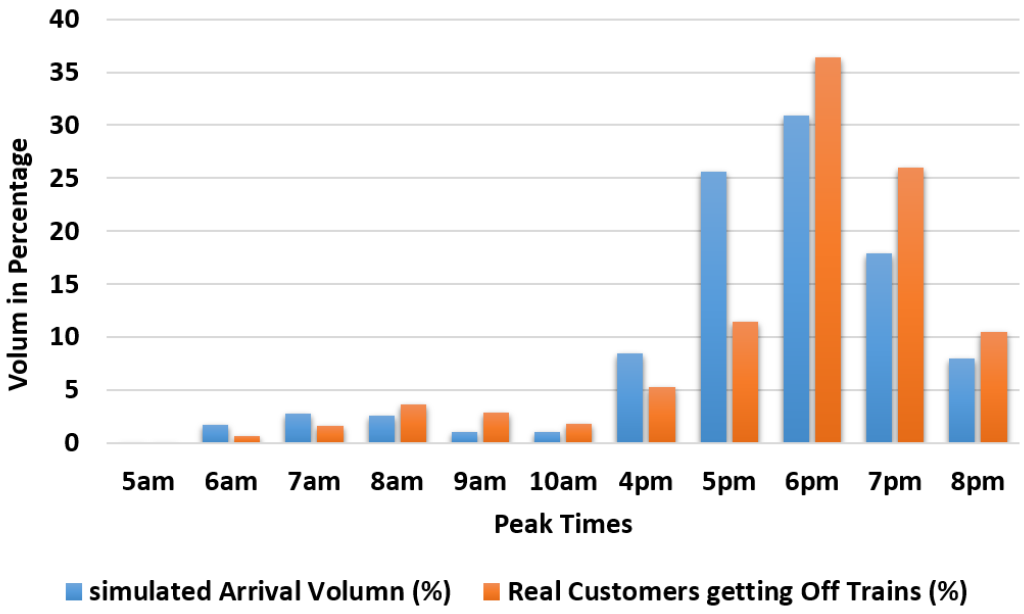

Figure 4.8: Comparing agent volumes arriving at all virtual station links at peak time to the count data volumes of customers getting off the trains at the same time.

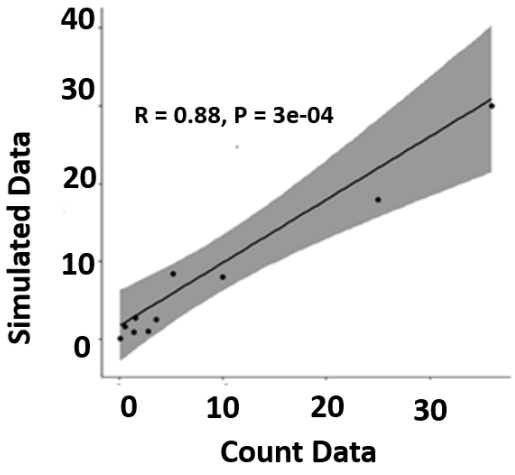

Figure 4.9: Correlation test between the volume of agent arriving at virtual train stops per time and the number of passengers getting off the trains in the count data

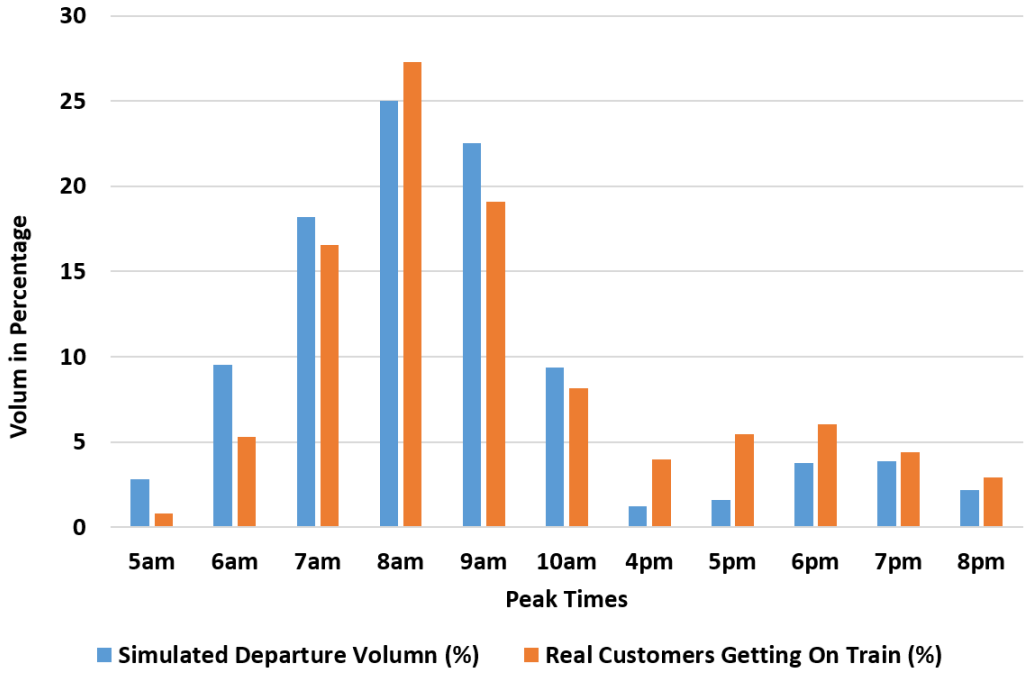

Figure 4.10: Comparing agent volumes departing from all virtual train station links at peak time to the count data volumes of customers getting on the trains at the same time.

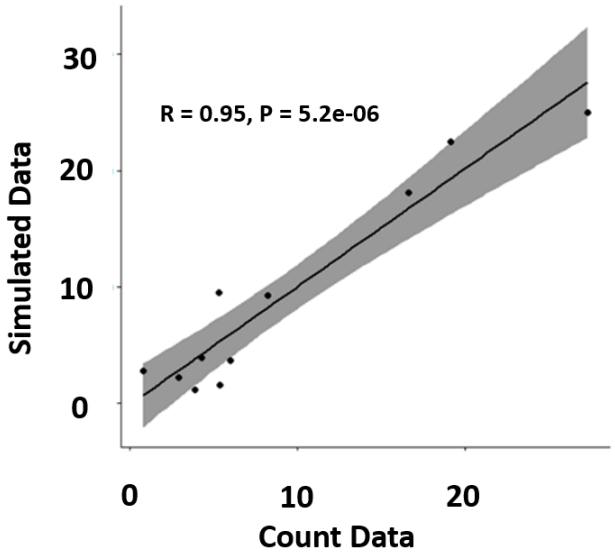

Figure 4.11: Correlation test between the volume of agent departing from virtual train stop links per time and the number of passengers getting on the trains in the count data.

4.3.12 Peak pricing policy analysis

We intend to see how some basic pricing policies used by PTO's will influence customer behaviour. We will be considering seven different fare pricing policies shown in figure4.12below.

Peak Surcharge: Fares at AM peaks and PM peak times.

Non mid-day surcharge: Charging base fares only at mid-day.

Mid-day discount: Fares are discounted at mid-day only.

Off peak Discount: Fare discount at early mornings, midday and evenings.

Differential fare increase: General increase in fares, especially during peak period.

Differential fare reduction: General reduction in fare, especially at off-peak times.

Peak surcharge and off-peak discount: Fare discount at off-peak times and fare increase at peak times

Morning fares for the MNR rails starts from 12am to 4:59, AM peak (morning peak) starts from 5am to 9:59am, mid-day is from 10:00am to 3:59pm, PM peak (evening peak) is from 4pm to 8:59pm while evenings starts from 9:00pm to 11:59pm. For the sake of analysis, we would assume that a flat price was in place at all hours of the day and that each customer bought a boarding ticket at point of entry.

By changing the fare price of each agent, different price strategies are simulated. For example, the peak surcharge policy is simulated by increasing the flat price by 40 percentage during the AM and PM peak.

Reducing demand peak All the policies applied reduce the AM peak by at least 9 percentage. The midday discount policy and the non-midday surcharge seem to produce the same effect, and reduce the am peak by up to 14 parentage, with a slight

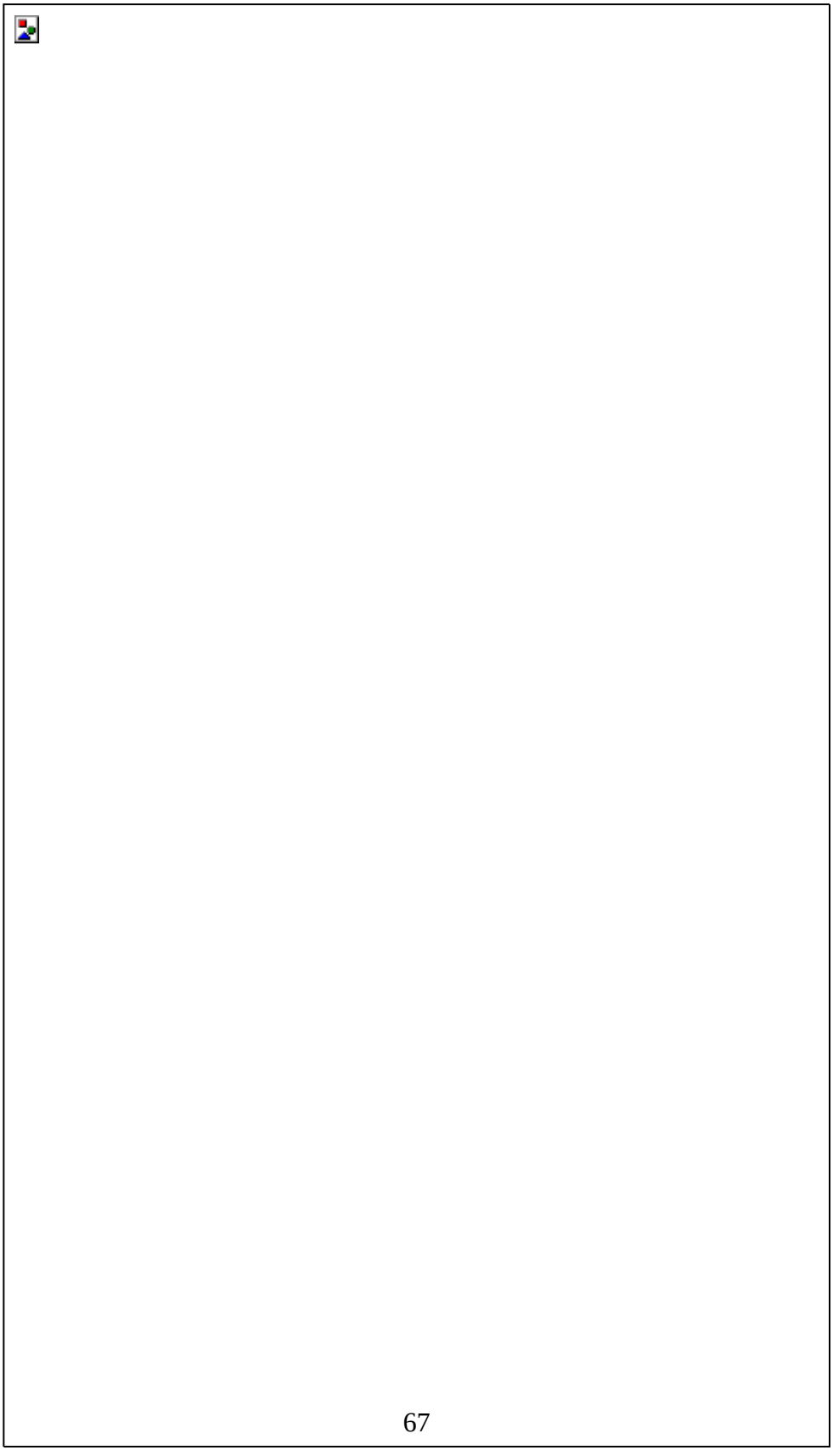

Figure 4.12: Common daily fare policy used by many PTOs

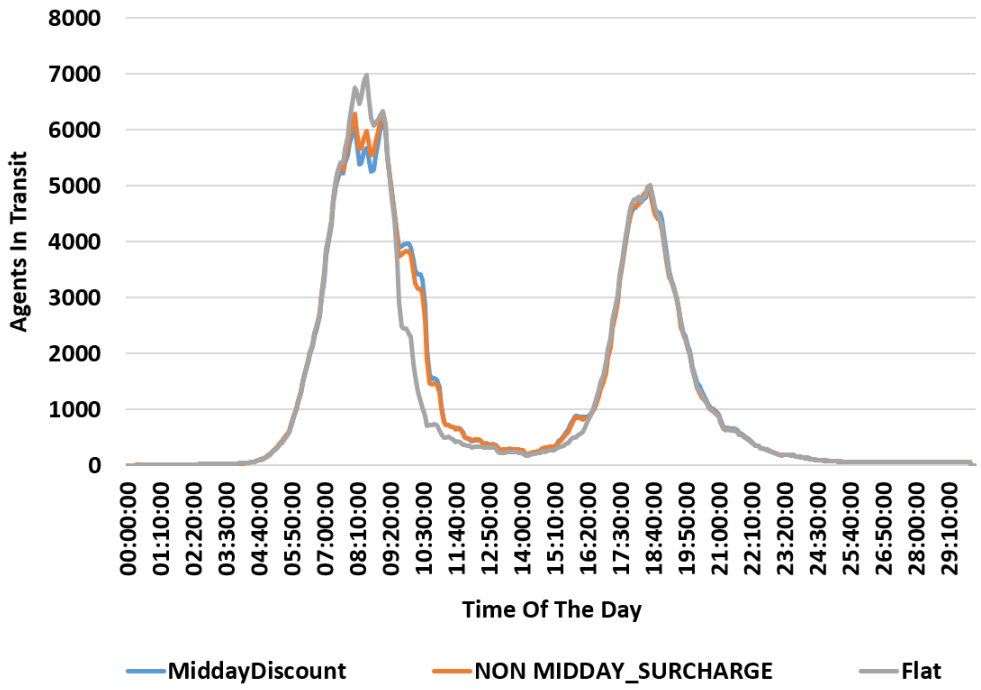

Figure 4.13: Mid-day Discount vs Non Midday Surcharge Vs Flat

widening of agents in the morning. This is common with other policies also. This could be the effect of agents with “recreation” and “school” travel purposes leaving a little later (or earlier) than normal to take advantage of price changes. The policies in figure4.14produce similar effects, reducing the evening peaks by about 16 percentage.

Shifting Peak time The policies in figure4.14seem to have the ability to shift the peak times. The AM peak shifted by more than 30 minutes while the evening peak was widened. Therefore a PTO that seeks such effect in demand might consider any of these.

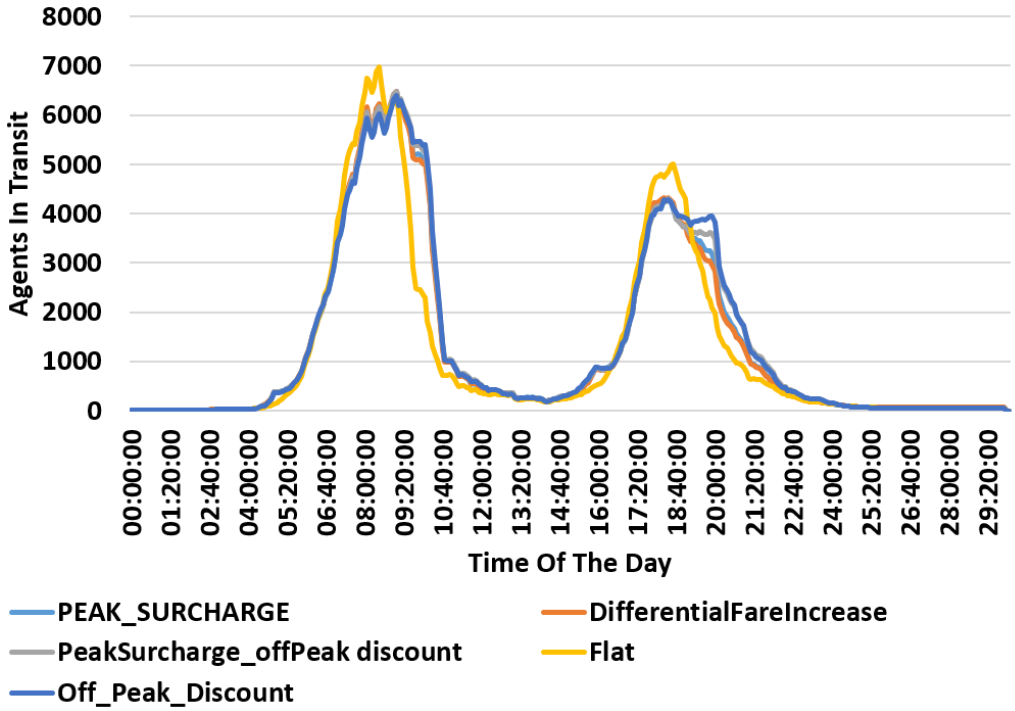

Figure 4.14: Peak Surcharge vs Differential Fare Increase vs Peak surcharge off peak discount vs off-peak Discount Vs Flat

4.4 Chapter summary and Future Works

This chapter has majorly discussed simulating a commuter rail network using the DDABMS framework. We used a customer survey data-set to create agent profiles which were then used in creating a combined agent population. The daily plans of the individual agents were optimized, as they utilized a simulated commuter rail system. The purpose of all this is to create a model that can serve as a test bed for testing policies. This test bed was validated and policy testing was demonstrated by running a simulation to forecast passenger demand behaviour on commuter rail system as pricing policies are applied in order to manage peak demand. We were able to determine poli-

cies that will likely generate similar behaviours (as shown in Figure4.14and4.13), and how different pricing policies have different effect on peak demand. All the data used in creating this simulation have come from open data sources easily accessible on-line, and our biggest challenge in creating this simulation has been in the accumulation of data.

In general, 70 percentage of the commuters of MNR are going to or coming from "work". This explains why the flexibility of their behaviour due to price change is not as pronounced, as it is difficult for such commuters to change their schedule or their route. By adding more municipality data, such as the spatial locations of businesses within the metropolitan area, we would be able to test the impact of adding other train lines, allowing the agents more choices in terms of route change. If we are also able to add the data of other modes such as buses, then mode shift impact analysis would also be a possibility, as accurate microscopic data of other optional transportation modes can be used in bridging the assumption made that customers will remain loyal after a price changes. With more data, a holistic view of the impact of a policy change, such as price change, to the rail system can be estimated. Also the GTFS files used in the simulation is for recent years as compared to the customer survey data. This is more of a data challenge as we were unable to obtain GTFS files as far back as 2007. But there have been minimal changes to the schedule of the PTO over the years.

Finally, as at the time the survey data-set was created, the PTO already had a pricing strategy in motion, therefore in our future works we aim to improve the model such that a predicted flat price can be simulated, which will be the basis for predicting and comparing the other pricing strategies.

In this chapter we have simulated a complete social system, using the data-driven agent based micro simulation (DDABMS) framework as detailed in Chapter 3 . The

success of this model further emphasises the importance of such data-driven individual level, granular models and the perfection of the methodologies in creating them.

In the next chapter, another published work is detailed, which uses the novel framework in recreated a distributed computer network using individual level data, after which policies are analysed.

Chapter 5

Simulating a Distributed Computer Network Using the Data-Driven Agent-based Micro-Simulation Framework

In the previous chapter we have applied the data-driven agent based micro-simulation framework, discussed in chapter 3, in predicting passenger behaviour in a passenger rail network. In this chapter we will further apply this data-driven agent based micro-simulation approach in modelling a distributed computer network. This approach is applied such that, each individual entity (i.e. individual users, workstations and servers) within the computer network is represented individually, so that every agent is a direct model of a corresponding entity (e.g. agent 1 is a direct model of user 1) within the computer network. The aim of this approach is so that, the interaction of all these micro agents will create a macro behaviour that will directly reflect the overall behaviour

of the distributed network. The creation of such overall model can then be used in creating ”what-if” scenarios that can be used for testing policies that network administrators would like to apply on the computer network in the near future. This work has been published in the FiCloud 2019 conference.

5.1 Introduction

Recent developments in distributed networks have heightened the need for network behaviour prediction [94]. A distributed network is a network system, over which computer programming, software and its data, are spread out across more than one computer, but communicate complex messages through various dependent nodes. Typically, in a distributed networking system the prediction of how these components interact with one another is vital.

The scale and the complexity of networks have dramatically increased in the last few years, with distributed networks becoming increasingly complex with the emergence of more distributed applications [102]. This complexity will continue to grow in the future with the rise of Machine-to-Machine communication, applications and other components running within a distributed network. It will become even more compelling and challenging to design scalable network traffic monitoring and analysis tools, which are able to handle complex interplay between network infrastructure and the traffic profiles generated by a continuously growing number of applications [1]. The current and future network monitoring frameworks cannot rely only on information gathered at a single network node, but, rather consolidate information from various network points and nodes distributed across the network.

However, in the last decade, systems have been designed and implemented for the

extraction of data from individual nodes within a computer network [76]. Despite the recent major advances of Machine learning and Big Data analysis frameworks, their application to the network traffic monitoring and analysis has received minimal attention. In addition, other critical applications such as detection of anomalies on a network, network attacks and intrusions requires an efficient, dynamic and fast mechanisms for real time analysis of thousands of events per second, as well as robust, and reliable techniques for offline analysis of massive historical data. Machine learning, agent based modelling and data mining based techniques are able to detect, characterize, and troubleshoot network anomalies and security incidents [89][92][26]. The application of these techniques will help in understanding and efficiently utilizing the amount of available network data [10].

Network operators and researchers are faced with various network and application challenge, and each network application also has its own features and performance requirements, which may change dynamically with time and space. Because of the diversity and complexity of networks, specific robust and dynamic machine learning algorithms is required for different network scenarios based on the network characteristics and user demands [55]. Therefore, developing a suitable and efficient algorithm and systems to deal with complex problems in different network scenarios remains a challenging task.

5.2 Aims, Objective and Motivation

The aim of this work is to propose a methodology that recreates the macro (overall) behaviour of a target distributed computer network, by creating micro agents from individual component of the system. We assume that the users within every network

have a probabilistic user behaviour pattern while accessing the system, leaving a behavioural trail that can be learned. This learned model is then used in creating user agent's behaviour. Agents representing other entities (such as servers) within the network, will also be created. These agents, combined with the individual user agents (modelled after the individual attributes and behaviour of each user) can communicate, thereby creating a game like interplay, which results in an emergent behaviour, similar to the target distributed computer network. This could serve as a test bed for predicting what-if scenarios within the network and for testing security policies and risk assessment on the target network.

The motivation behind the application of data-driven agent-based micro simulation and machine learning(ML) techniques in a typical distributed computer network environment is for the following reasons:

Firstly, the best-known capabilities of Machine learning techniques such as classification and prediction plays a vital role towards solving networking problems such as intrusion detection and performance prediction in distributed networks.

Secondly, agent based and micro simulation models are efficient in creating decision making tools, which would facilitate the prediction of the outcome of scenarios that cannot be tested on the target system.

The rest of this chapter is structured as follows. Section5.3gives an overview of the related work. After which an overview of how the DDABMS concept is used in this case study is explained. The learning behaviour pattern of each user and the base line experimental results is also presented,then the validation of the simulated model is discussed. Finally we then use this base line model to test the vulnerabilities within the simulated distributed system. Further works and conclusions are made in the final section.

5.3 Related Work

The behaviour and control of a communication network is an important aspect in every distributed network. The infrastructure of a distributed network produces huge amount of unstructured and unlabelled information, (both as very short-lived data), such as link statistics, as well as trend data, such as bandwidth utilization over time. ML methods have been successfully used to discover optimized solutions to this challenge and have the capability to learn the network behaviour under different network conditions and operation.

In recent times, a considerable amount of literature has been published on network behaviour and communication network control. These studies have mostly worked on network behaviour using aggregated data compared to our proposed model where individual level data is utilized.

One of such authors: Moore et al [70] applied supervised Naive Bayes technique to classify network traffic by application, based on packet flow statistical features. Although the result was able to provide 65 percent accuracy for data from the same period and can achieve over 95 percent accuracy when combined with a number of simple refinements, the primary data used in training the ML model was aggregated. Similarly, William et al[106] presented an evaluation of several classification algorithms, including Naive Bayes and C4.5, to classify IP traffic flows. They defined 22 practical flow features for use within IP traffic classification and further reduced the number of features using correlation-based and consistency based feature reduction algorithms.

Studies conducted by Arroyo-Valles et al. [4] proposes QProbabilistic Routing (QPR), which is an energy-aware routing algorithm for wireless sensor networks which incorporates intelligent routing decisions adapted to the network dynamics by using a

cost metric locally learned from previous neighbour interactions and taking forwarding decisions by means of a Bayesian classifier, with simulation results showing that Q-PR allows to increase successful message transmissions as well as network lifetime. Khurum et al. also applied unsupervised machine learning methods on data extracted from cyber physical system of a water treatment facility to predict cyber-attacks on the facility with low false positive rates and high precision and recall [41].

A number of researcher have also conducted various surveys on ML in networking [2][12][15][28][52][75]. Such that while the authors in [75] addressed traffic classification in networking the authors in [28][15] and [15] have applied ML to specific problems in networking. On the other hand,[52][12] and [2], though comprehensive in their coverage of ML techniques in networking looked at specific network technology such as cellular network and wireless sensor network.

Also, building behavioural models from network user data, with ML algorithm has also been well studied, with research work such as Hlosta et al.[58] who extracted online behavioural pattern from student in a virtual learning environment using a Markov chain based analysis with the intent of finding a superior method to analyse student behaviour, to find the causal elements of student failure.

Behavioural models have also been used in creating malware mitigation software by using behavioural extraction techniques that constantly record and learn user patterns in order to create malware detection systems that would need minimal Modifications to mitigate malware threat in android device [96].

Following the above related works, we propose a methodology for creating more realistic predictions of distributed computer network behaviour by using machine learning and agent based micro simulations. This methodology follows the DDABMS process discussed in chapter 3.

5.4 Simulation Methodology

5.4.1 Theoretical Model

Following process 1.0 in chapter 3, a theoretical model of the whole simulation is created to highlight the idea, direction and process of the simulation. Firstly, in this simulation, we intend to simulate the way users make requests for the network resources(e.g, software applications), and the way the network reacts in fulfilling those requests. The intent is that, by doing this we would model the whole behaviour of the target system, thereby creating a test bed for policy testing. There are three major entities we identify in a computer distributed network, and hence intend to model. These include:

1. User: The human individual using the resources of the network for his private project.
2. Workstation: The client side computer being used by the user.
3. Server: The server computer hosting the network's application and serving the workstations within the network.

For simplicity, we assume that each user has his own personal workstation. Therefore, we intend to model the user and his workstation as one agent. The following are the generic steps in applying DDABMS in simulating a distributed computer network:

1. Track individual user online data on a target distributed network.
2. Apply machine learning on individual usage data sets so as to extract individual behavioural patterns. This pattern is used in creating individual user agents, whose behavioural pattern reflect that of the users they represent.

3. Creation of agent representation of other elements such as server's within the network using rule based behaviour.

4. Initialize individual agent attributes using the direct initialization method (see chapter 3).

5. The communication behaviour of all agents is simulated to create an emergent behaviour similar to the target system. The emergent behaviour is then validated by comparing the simulated request logged by the server agent to the server logs of the target network.

6. The validated base line model serves as a test bed for testing different scenarios associated to the network. One of such scenarios, as discussed in subsequent section, is a virus attack scenario where the network vulnerability is discovered in the event of a simulated attack, allowing for a simple risk and vulnerability assessment of the target network using the simulated network as a test bed.

These steps can be further summarized into a flowchart with four process stages, as shown in Figure5.1below:

1. Data source process: behaviour and attribute data is extracted from network users at client side and from network servers.

2. Input process: The extracted data is transformed into machine readable format.

3. Core simulation: The behaviour data extracted from the user is mined for patterns (so as to create the agents), using a Bayesian network. To do this, a CPT (conditional probability table) of the user behaviour is first developed. Once this is done, a Markov chain Monte Carlo algorithm is used to simulate the user agent's behaviour towards the server agent.

4. Output process: the baseline model produced in the core simulation is validated. If this model reflects the behaviour of the chosen target system, we go ahead and simulate chosen scenarios. To validate the model, a comparison is made between the server logs in the target system and the simulated request made by the network users. *A* strong correlation between the simulated emergent behaviour and server logs will give insight to how accurate the predicted model is.

Figure 5.1: This is a theoretical model overview of the simulation, using a summarised flowchart, created to summarize the whole simulation project in a pictorial format. It also shows the extendable nature of the simulation, such that it covers all distributed computer networks, not withstanding their complexity. We show the four core modules involved in this joint process. It also shows the granular level focus of the DDABMS framework

5.4.2 Data Source

Following the framework as represented in subsection 3.1 in chapter 3, we discuss how we followed process 2.0 in establishing our data sources. In using DDABMS approach, three major data sources are important in this model. Firstly, the user data from every user (i.e. $user1$ to $userN$) within the network, containing the activity track of each user and the time indulged in this activity. Secondly, the system characteristics and attributes of all the work stations within the network, including the operating system, installed applications and system availability. Thirdly, the server logs of application requests, stored within the network server's operating system, containing a history of requests made to the server from the individual workstations which will eventually serve as the validation data (i.e. the data to be used for validating the simulation).

5.4.3 Agent Creation System

This subsection, further discusses how the agent creation system (i.e. process 3.0) was used in creating an agent population, with each agent reflecting each unique user or server. This then cumulatively represented the resource request system in the distributed computer network.

Acquire and Prepare data: following Process 3.1 in figure 3.2, the data set acquired for this simulation were static files download from each computer in our network. Our data collection consisted of a daily user track automatically collected by the ProcrastiTracker software [78] which tracked applications or documents utilized on the workstation and the time this activity was executed. A sample of such tracking activity for a single user can be seen in figure5.2and a sample behaviour raw data extracted from user 1 on day 1 of the experiment is seen in figure5.3which is then transformed

for machine readability in figure5.4.

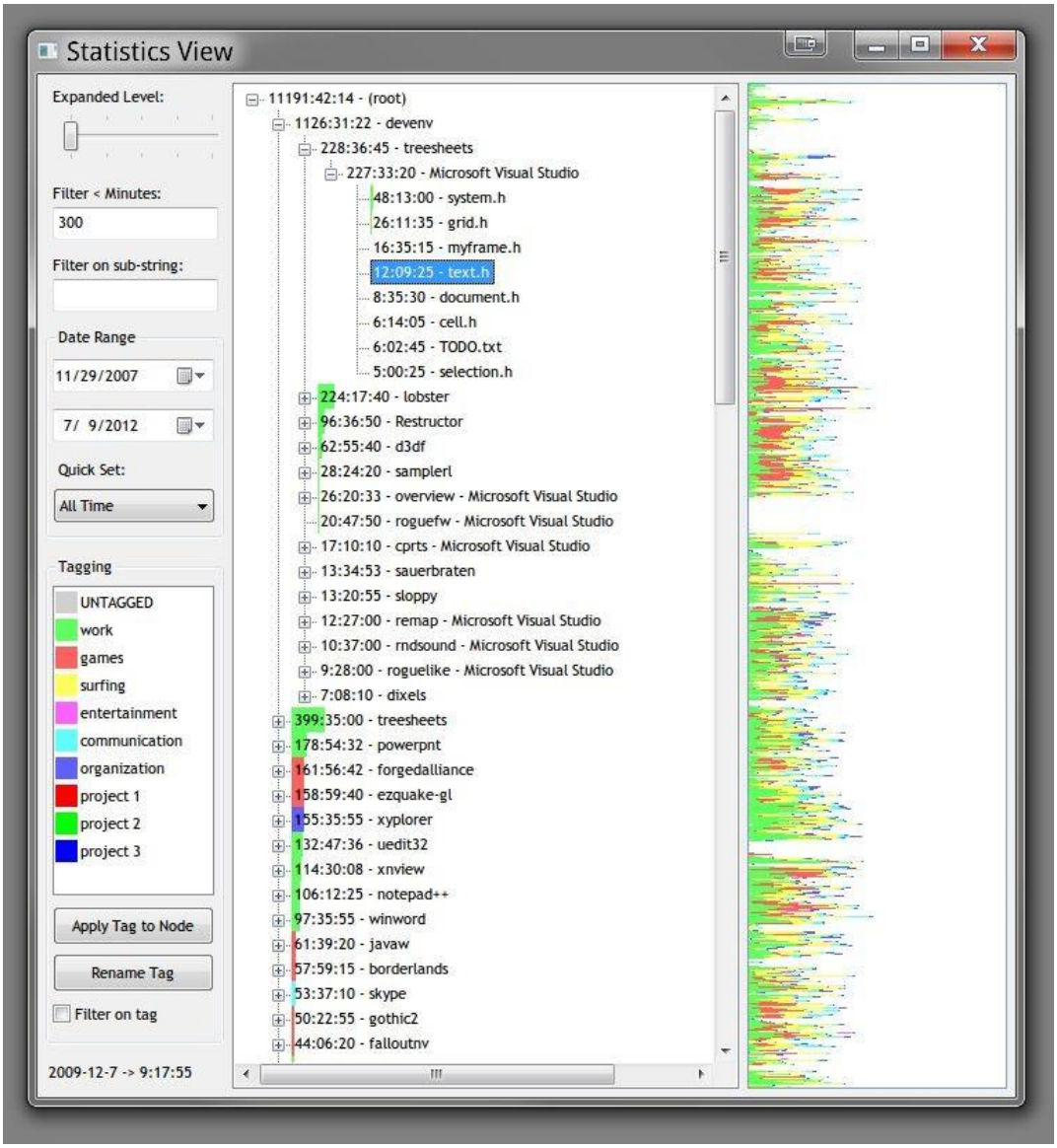

Figure 5.2: A tree view of the user tracking software, collating the user behaviour data.

07:05 prevhost - 11% of parent, 1 keys, 40 lmb, 1 rmb, 32 scrollwheel

- **45** xampp - 36% of parent, 1 keys, 13 lmb, 32 scrollwheel, 08:29 start on 2019-3-26
- **20** ProcrastiTracker - 16% of parent, 8 lmb, 08:27 start on 2019-3-26
- **20** Program Files - 16% of parent, 2 lmb, 08:26 start on 2019-3-26
- **20** images - 16% of parent, 8 lmb, 1 rmb, 08:27 start on 2019-3-26
- **10** Local Disk (C:) - 8% of parent, 4 lmb, 08:26 start on 2019-3-26
- **05** Program Files (x86) - 4% of parent, 3 lmb, 08:27 start on 2019-3-26

01:15 chrome - 6% of parent, 14 lmb, 12 scrollwheel

- **35** youtube.com - 46% of parent, 5 lmb, 5 scrollwheel
  - **25** "My Worship" - Phil Thompson (OFFICIAL) Session Recording - YouTube - Google Chrome - 71% of parent, 2 lmb, 14:32 start on 2019-3-26
  - **10** YouTube - Google Chrome - 28% of parent, 3 lmb, 5 scrollwheel, 14:32 start on 2019-3-26
- **10** New Tab - Google Chrome - 13% of parent, 3 lmb, 14:32 start on 2019-3-26

40 dllhost - 3% of parent, 14 lmb

- **20** statistics.jpg - Windows Photo Viewer - 50% of parent, 3 lmb, 14:27 start on 2019-3-26
- **05** statistics_small.jpg - Windows Photo Viewer - 12% of parent, 3 lmb, 14:28 start on 2019-3-26
- **05** settings.jpg - Windows Photo Viewer - 12% of parent, 3 lmb, 14:28 start on 2019-3-26
- **05** popup.jpg - Windows Photo Viewer - 12% of parent, 3 lmb, 14:28 start on 2019-3-26
- **05** awaydialog.jpg - Windows Photo Viewer - 12% of parent, 2 lmb, 14:28 start on 2019-3-26

30 xampp-control - 2% of parent, 9 lmb

- **25** XAMPP Control Panel v3.2.4 - 83% of parent, 7 lmb, 14:34 start on 2019-3-26
- **05** Configuration of Control Panel - 16% of parent, 2 lmb, 14:35 start on 2019-3-26

30 httpd - C: - xampp - xampp_start.exe - 2% of parent, 3 lmb, 1 rmb, 14:33 start on 2019-3-26

10 notepad++ - C: - Program Files - Notepad++ - change.log - Notepad++ - 0% of parent, 4 lmb, 14:31 start on 2019-3-26

05 xampp_start - C: - xampp - xampp_start.exe - 0% of parent, 1 lmb, 14:34 start on2019-3-26

Figure 5.3: This is a snapshot of a raw user file, showing the usage data of user 1 on a chosen day 1.

Day	t1	t2	t3	A1	A2	A3	A11	A12	A13	R
day1	1	0	0	0	0	1	0	0	0	1
day1	0	1	0	1	1	1	1	0	0	1
day1	0	0	1	0	0	0	0	0	0	0

Figure 5.4: This is the transformed data of the raw user data shown in figure5.3above

5.4.4 Model Individual Agents

Following figure 3.2, process 3.4, we classify each entity in the network into an agent class. The three major classes of agents have been stated in section5.4.1. They include: the server, users and workstations. The workstations and the users will be modelled as entity,as we assume the user is restricted to just one workstation.

To create a model of the user, a machine readable format of the online activity data of the user has to be created, It should contain both the user's attributes and behaviour. The purpose of the transformation is to allow the application of machine learning algorithms on the user data, and allow easy validation using the server log files. The user activity can be summarized as a tuple:

$$X = (T, R, A) \tag{5.1}$$

Where: T is the total day time, which the users are able to use the computer network. We divide this into time slots, for easy processing, such that: $T = [t_1, t_2, \ldots, t_n]$. A is the list of applications installed on the server and clients side of the network which users have access to. $A = [A_1, A_2, \ldots, A_K]$, where k is the total number of applications installed within the network and can be accessed by the workstation user. R is a request to log into the server from the client system of the user. Applications within A can have

sub elements such that $A_p = [A_{p1}, A_{p2}, \ldots, A_{pj}]$ where A_p can be any application A within the network and j is the total sub application within the application A_p.

Attributes specific to the target network are also extracted so as to initialize the agent environment within the simulation and enhance the creation of rule based behaviour of other agents.

5.4.5 Behavioural pattern recognition

Following Process 3.4.4 in figure 3.3, to extract behavioural patterns of agents, we use a machine learning algorithm. In choosing the most efficient machine learning algorithm, a number of factors were considered:

1. Human behaviours are generally probabilistic

2. The data extracted from a user might be minimal if the workstation is new or the user is new. Therefore, prior knowledge of the user behaviour might be needed.

3. Activities indulged in by the user can be considered as a chain of events such that the present event has a causal effect on the next event. Therefore, a generic chain of event (or activities) will be a user logging into a workstation at time t and makes a request R to the server for an application A_2 and/or A_3 while opening a local web application A_1 which access a site with risk value A_{11}, A_{12} or A_{13} (Where A_{11},A_{12} and A_{13} are risky, moderately risky and highly risky websites respectively). The event tuple can then be ordered as a chain of causal events which can be represented as a DAG $G(X, E)$ where $X = [x_1, x_2, \ldots, x_n]$ is the chain of events with n being the number of events and x being each activity. E is the list of edges connecting each activity to its subsequent one as shown

in figure5.5(the Conditional probability table of each node as extracted from the data can be found in appendixB). This can be represented as the Bayesian network, such that the joint probability of all the nodes is a combination of the conditional probabilities of individual nodes, as represented in equation5.2

$$P(x_1 \ldots x_n) = \prod_{i-1}^{n} P(x_i | x_1 \ldots x_{i-1}) \tag{5.2}$$

If we then view the transition from one event to another within the network as a state change, then this graph can be converted to a Markov chain [23] as represented in equation5.3:

$$M(X, E, T_{e \in E}) \tag{5.3}$$

Where T_e Is the transition kernel and e is the edge between two nodes (events). By calculating the individual probability table of each node, we can randomly walk through the Markov Chain and generate samples which will eventually converge to a target posterior distribution using a simple MCMC simulation.

5.4.6 Create Agent

To create a user agent, each individual behavioural model is created by applying individual data-set to the DAG, to create a Markov Chain. When this is then coupled with the individual attribute of each user, then rules for resource request from the user to the server and resource access grant from the server to the agent can then be coded.

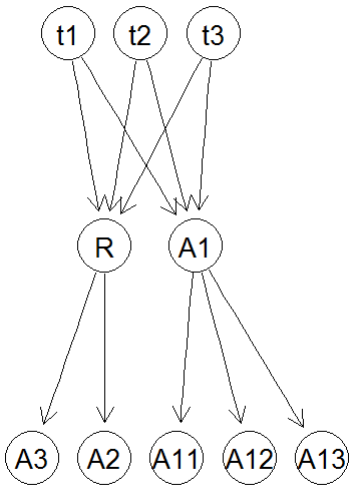

Figure 5.5: DAG representing the chain of events/activities of a user when logged into workstation at time T

5.4.7 Scaling

For the scaling process 5 in figure 3.1, this simulation was not scaled down or up, but was the exact size of the model (i.e, 1:1). This implies that the data extracted from every entity in the real network was directly used in creating an agent in the modeled network.

5.4.8 Core simulation

At the core module, the Directed Acyclic Graph (DAG) model created for each user is simulated using a Markov Chain Monte Carlo (MCMC)[23] simulation. The simulation is such that, each node in the DAG model of the user is activated (i.e, 1) or inactivated (i.e, 0) based on the Monte Carlo seeds at every round. This then determines the decision of the user agent to either request for a particular resource/application or not.

5.5 Running the Baseline Simulation

In order to prove the proposed concept, we use a simple distributed network consisting three workstations (for 3 users) and a windows server running on windows server 2016 operating system (OS) hosting two applications: A_2 (Notepad++) and A_3 (xampp), which were constantly requested for by the users. In the course of our data collection, we restricted users of the network to specific workstations so as to enhance easy data gathering directly from the workstations. Each user workstation allows unrestricted internet access through a browser A_1 where $A_1 = [A_{11}, A_{12}, A_{13}]$ and: A_{13} are high risk websites, A_{12} are moderate risk websites and A_{11}represent normal websites. A pictorial representation of the distributed network's architecture is given in figure5.6 below.

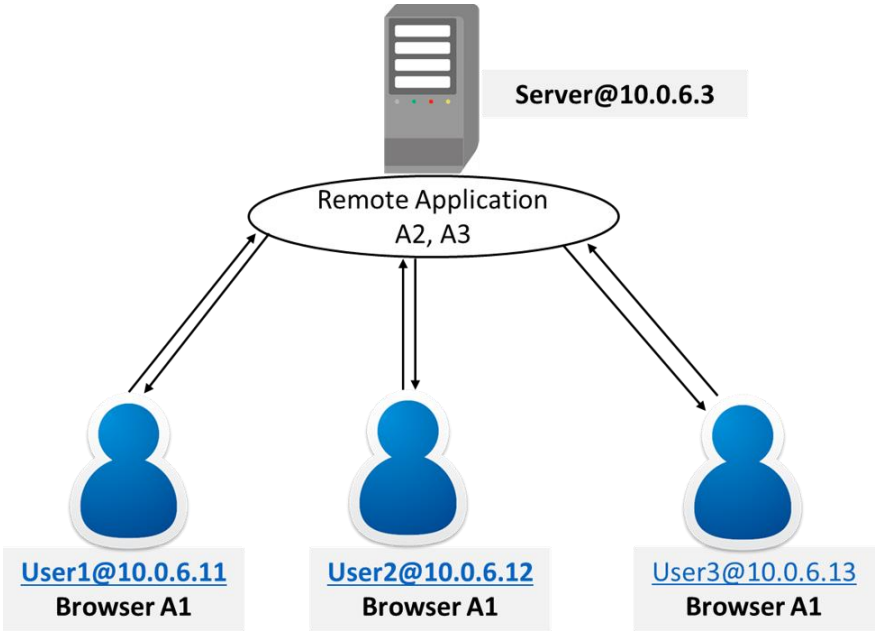

Figure 5.6: A simple distributed network architecture used in testing the methodology

The server log files were directly retrieved from the server. The user activity was

observed in three time slots: morning slot t_1 (4am to 11:59am), afternoon slot t_2 (12pm to 7:59pm) and night slot t_3 (8pm to 3:59am). This is represented as a 1 if the workstation is used within the observed time slot, and a 0 if otherwise. Each individual workstation usage track data set was then converted from its original html (as seen in figure5.3and5.4) format to an $n \times m$ matrix where n is the number of activities considered for observation in every users activity profile (this we refer to as the user attributes) and m is the number of time slots multiplied by the number of days in which the data was observed. A snapshot of this matrix for user 1 on day 1 of the observation can be seen in figure5.4. This snapshot shows that application A_2 (notpad++) was requested from the server at t_1 (in the afternoon) while A_3 (Xampp) was requested at t_1 and t_2 (in the morning and afternoon). A_1 (browser) was used in the afternoon and a low risk website (A_{11}) was visited. The architecture of our distributed network is shown in figure5.6below.

The simulation was executed on Intel i7, 3.6GHz processor and 8GB RAM computer with windows 10 as the base operating system. 10,000 iterations were simulated, to bring the simulation to equilibrium, where each node eventually collapsed to the limiting distribution which is our mean. To check for equilibrium, we view the number of variations from the mean in the latter distribution of each node using a simple z-score plot. Figure5.7displays the Z-score plot of the iterations for the simulated request (R) from $agent_3$ (created from behaviour of $user_3$) to the server agent while figure5.8shows a similar plot of the same agent for application A_1.

Since the agent uses the MCMC algorithm to decide weather to make a request for an application from the server agent or not, and the probability of making such request has been calculated, based on the historical online behaviour data of the actual user, the Z score is a simple check that shows if the probability of the user agent

requesting a resource from the server agent, is the same as the probability of the real user requesting the same resource from the real server. If the observation points of a Z plot oscillates around 0 (as seen in Figure5.7and5.8), this shows that these two probabilities are similar. Therefore, we can see from Figure5.7, that in the course of the simulation iterations, the difference in the probability of the user making a request for applications at a particular time to the server, and the difference in the probability of the corresponding agent, making a request to the server agent applications at that same time is close to 0. Figure5.8is also similar to Figure5.7, except that we focus on a particular application A_1.

In the next sub section, the simulation is then validated by comparing the simulated network data with the real life data from the target network.

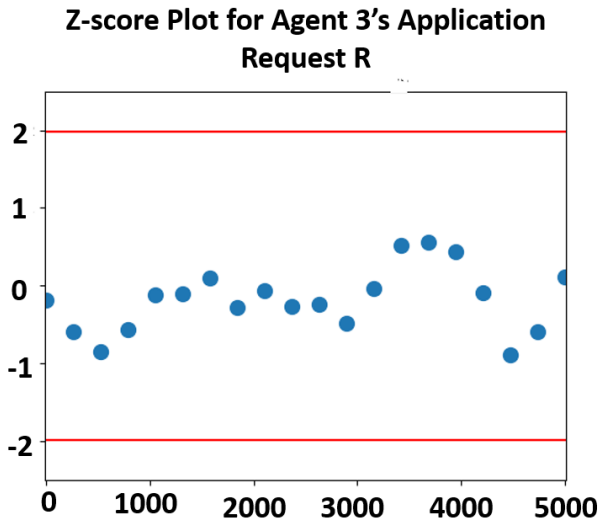

Figure 5.7: The Z-score plot. This shows the difference in the probability of a user making a request (R) to the server at any time of the day, to the probability of the user's corresponding agent making a request to the server agent at the same time of the day, is always close to 0. This is why the observation's in the graph keep oscillating around 0, in every iteration of the simulation.

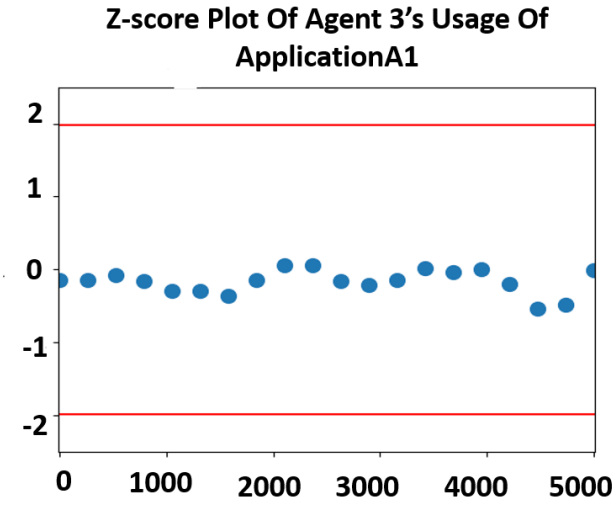

Figure 5.8: The Z-score plot. This shows the difference in the probability of a user using the application A_1 at any time of the day, to the probability of the user's corresponding agent using the same application at the same time of the day, is always close to 0. This is why the observation's in the graph keep oscillating around 0, in every iteration of the simulation.

5.6 Validation

I addition to the data collected on every user, the request for resources made by every user, to the server, is also stored by the server. This is also collected and used as a validation data. This validation data is compared to the resource request made by the agents in the simulation. The degree of similarity between these two data sets will serve as a form of validation for the simulation.

This validation is done using a correlation test, and was performed between the simulation request by the user agents to the server agent, and the real request made by the real users to the server (as documented in the server log). We recorded a correlation co-efficiency of 0.89 as shown in figure5.6.

We do not expect a perfect correlation between the two data sets, as there could be a time lag between the actual time the user makes a request for an application from the

server and the time it actually opens at the server end, especially if the request is made very close to the end of a time slot. In such a case, the request might be made at a time slot t_1 (e.g, 11.59am) but the request might be recorded on the server at time slot t_2 (12:01pm).

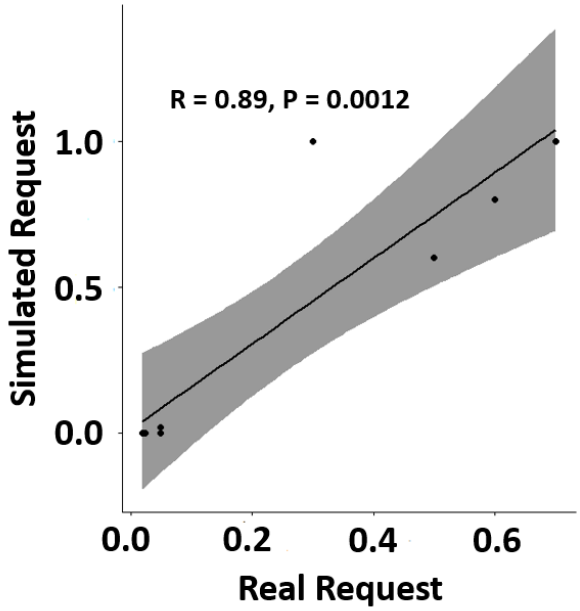

Figure 5.9: Correlation test between real and simulated requests

5.7 Testing The Vulnerability Of A Network Using Alternative Scenarios

Based on the learned pattern, we examine the vulnerability of the network by allowing the user agent be infected with virus when they visit a highly risky website (A_{13}) through the application A_1. An infected user agent will have the ability to infect the server if a request is made to the server from the user's workstation, and an infected server can infect a client work station if a request is made by a non-infected work-

station. The validated simulation can be used as a test bed for the target network to test its vulnerability to future attacks so as to aid risk analysis and decision making for network security. We simulated 3 scenarios of 1000 iterations with user agents communicating with server agents to request applications hosted on the server while also accessing internet websites. In the first scenario, network communication at time slot t_1 (morning) was simulated, with results shown in figure5.10In the second scenario, network communication at time slot t_2 (afternoon) was simulated as shown in figure 5.11. In the third scenario network communication at time slot t_3 (night) was simulated in figure5.12, while in the final scenario the network communication of the whole day was simulated as seen in figure5.13. All with the intent of revealing vulnerabilities based on usage patterns. For each user agent in the simulated network, we compare the frequency of request made to the server agent by the user agent to the number of iterations passed before the user agent's workstation was attacked by a virus.

1. *Morning*: It can be seen from figure5.10that $user1$ (i.e. user agent 1) makes very few request to the server and is quickly infected by a virus (based on the few iteration counts before user 1 was infected). $User2$ is quickly infected by a virus and makes frequent request to the server. $User3$ makes few request to the server and is not easily infected. In retrospect, this shows that in the mornings $user1$ and $user3$ are least likely to make request to the server, but if $user1$ does make a request in the morning he is likely to go to a high risk website and put the server at risk eventually compromises the network as compared to $user3$ whose usage pattern, though scanty in the morning is less likely to get a virus

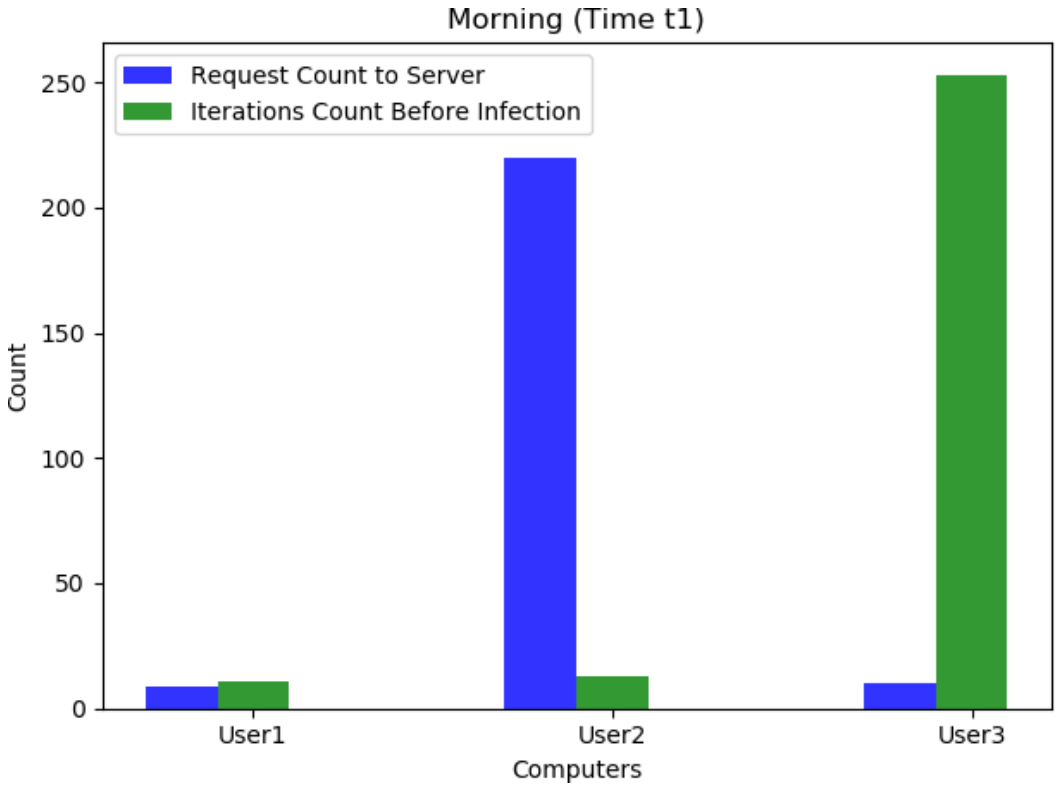

Figure 5.10: A scenario in which an attack comes in at time t1

2. *Afternoon*: figure5.11shows the simulated network at time t_2 (afternoon). Similar to the morning scenario, $user2$ is very active in terms of server request and also prone to attack which would easily make the user a gateway to infect the server. $User1$ is less active in the afternoon unlike morning but is still prone to attack and can be a gateway to infect the server. $User3$ on the other hand has the most few request count to the server making this user the least active in the afternoon and the least to get a virus attack based on the number of simulation iterations before being infected.

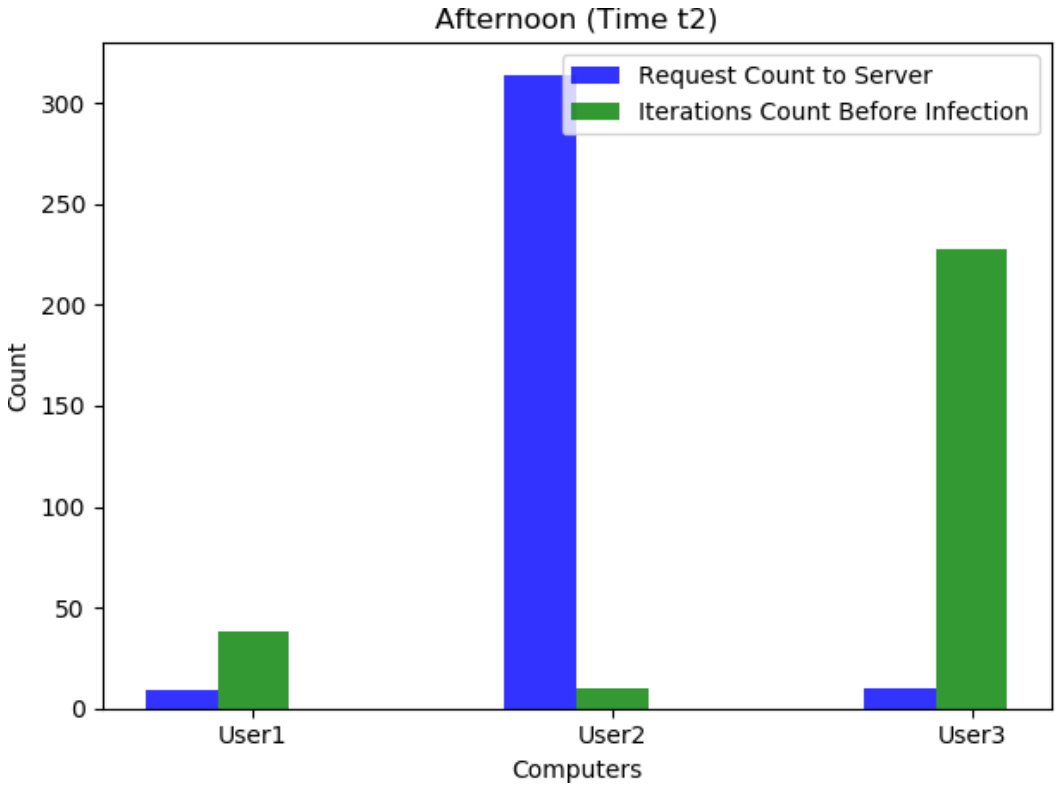

Figure 5.11: A scenario in which an attack comes in at time t2

3. *Evening*: At time t_3 (evening), figure5.11shows that $user1$ becomes active, even more than $user2$ who has been the most active at previous times as his server request surpasses that of $user2$. The user is also as prone to being infected from website viruses as $user2$ thereby being the most likely to compromise the network at time t_3. $User3$ still maintains low activity leaving $user1$ as the weakest link at time t_3.

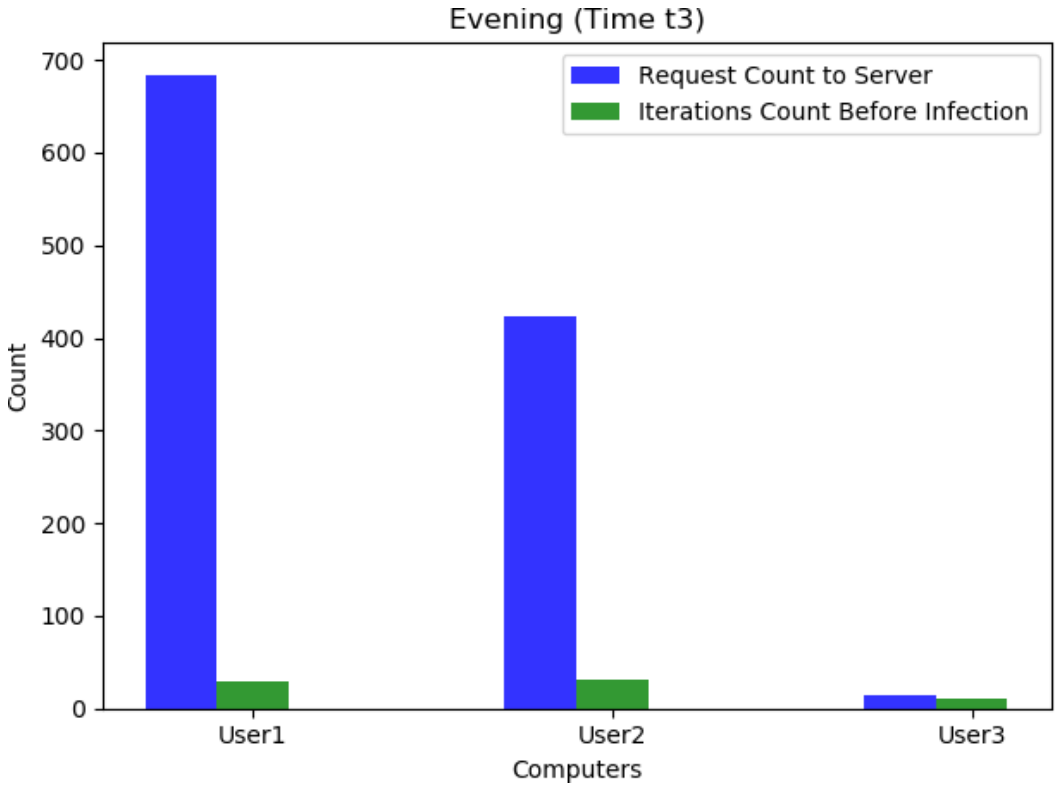

Figure 5.12: A scenario in which an attack comes in at time t3

4. *All day*:By simulating the interactions of the workstations and server within the network indiscriminately at time t_1, t_2, and t_3, an emergent behaviour represented by figure5.13emerges revealing $user1$ and $user2$ as the weakest link in the network and the most likely gateway for a viral attack on the server which would eventually compromise the network. It also reveals $user2$ makes more request to the network than any other user and $user3$ is the least active user on the network and also the least likely to be vulnerable to attack or compromise the network. Not just because of his scares request to the server but because of his behavioural pattern which has the least probability of exposure to high risk websites.

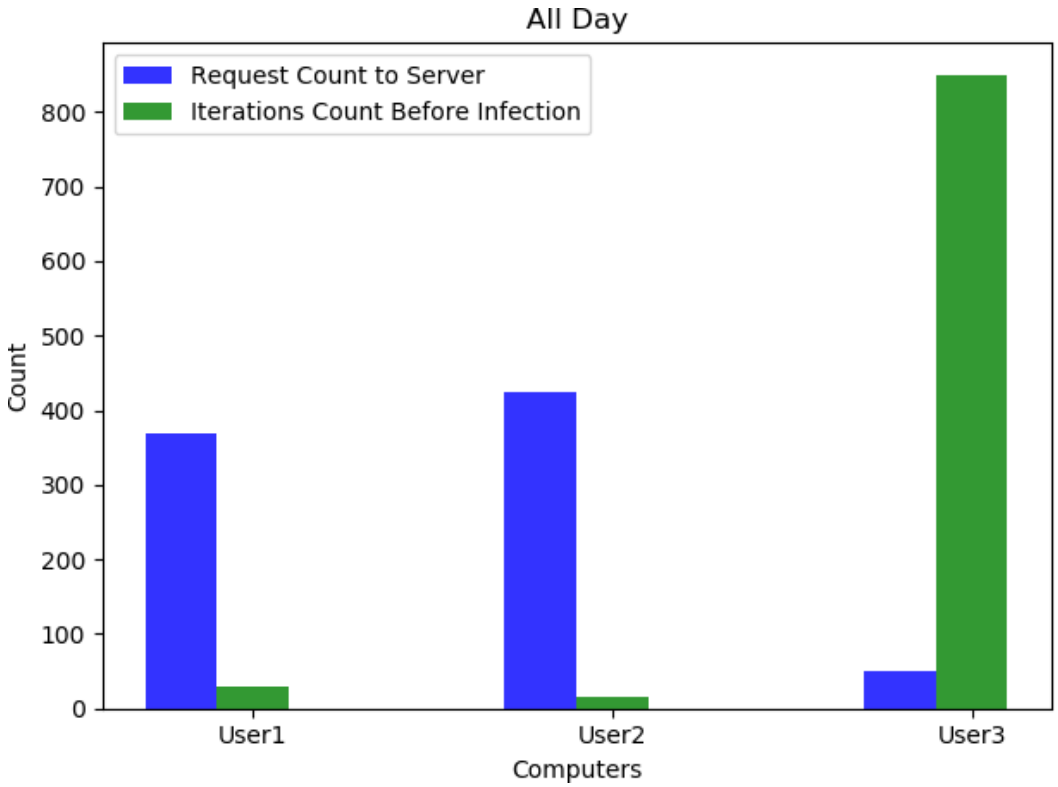

Figure 5.13: General vulnerability of the network

5.8 Chapter Summary and Conclusion

In this chapter a DDABMS application case study for simulating the behaviour of a distributed network, using a bottom up focus, was the major points of discussion. Individual behaviour computer network user's online behavioural pattern was captured and predicted using a machine learning algorithm. This was further used in creating corresponding individual agents, empowered with the behaviour pattern of actual users. A distributed network environment was simulated using these agents, which then produced an emergent behaviour similar to the real distributed network, with a correlation coefficient greater than 0.8. This baseline model was used in testing the vulnerability

of the real system.

This test bed created from this simulation was used in testing which user's work-station, within the computer network, is most likely to be the first, within the network, to be infected by a virus, external to the network. The probability of this event happening through each of these agents, in the course of the simulation, sheds light on how vulnerable the corresponding user was, and how much danger he poses to the system. The four scenarios tested, allowed use to investigate the vulnerable users within the network, at different times of the day.

This same simulation can, in the future, be used in investigating load balancing within a computer network. This can be easily implemented, since the request pattern of the user agents to the agent server, aligns with the request pattern of the actual user to the actual server. This implies that the load on the agent server will be the same as the load on the real server. Further more, in future we hope to expand this work by using big data tools to accommodate the simulation of more users.

The success of this model further emphasises the importance of data-driven individual level granular, models and the perfection of the methodologies in creating them. This case study also shows the generic strength of the DDABMS frame work discussed in chapter 3, as this is a totally different social system from that discussed in chapter 4.

In the next chapter we extend the DDABMS model to accommodate dynamic data, making it adaptable in the face of constantly changing recordings from the targeted social system.

Chapter 6

A Dynamic Data Driven Agent-Based Micro-Simulation Framework

Data-driven agent based micro-simulations (DDABM) have been applied to various domain problems, as seen in previous chapters. These chapters have shown how unique individual actors, within a target system, can be modelled directly into agents, by extracting attributes and behavioural patterns from individual level data. Although, these methods have created more accurate models, that have better reflected the social systems they represent, these models, when used in predictive analysis would face some limitations in the long term. Since the simulation models usually abstract the target system at a certain time, there will exist an error between the simulated reality and the target system in the future. This error might be minute at the point of the model creation, but as time passes, the error accumulates, thus negatively affecting the reliability of the result. This is mainly due to the fact that human behaviours are subject to change. And also, the environment around the human subject, in a social system, can also change, or can instigate behavioural change. This will thereby, annul the attributes

and behavioural patterns extracted from the historical data previously gathered. Take for instance, the passengers (travellers) simulated in Chapter4. If in a few weeks, after the simulation has been created, the travel behaviour pattern of a traveller changes (e.g. if a traveller gets a new job that requires him to take a different route or stop at a different station), a response to such adaptability is not accommodated in the simulation, thus allowing a continuously increasing inaccuracy in the model as time passes.

This challenge is not new in the simulation of physical systems [93], but has been surmounted, due to the availability of real time data-monitoring sensors [61][65]. Such availability of data, allows the continual measurement of parameters, thereby allowing the development of adaptable and dynamic models through consistent calibration. Such data sources have also become available for social systems, because of the availability of different IOT (internet of things) and data collection tools, that help collect individual level data from which human behavioural patterns can be extracted.

For physical systems, the adaptation/calibration of the model with dynamic data (i.e, consistently changing data), is mainly done by tuning the model parameters, while comparing the simulation result to the newly measured data, from the target system. According to [6], this approach would be highly limited if applied to agent based models of social complex systems, due to the fact that:

1. Agents are autonomous, and ABM's are interaction based, and therefore, meant to be uncontrollable.

2. Sometimes, it is difficult to have a solid grasp of the complex system the model is being applied to, thereby making it difficult to generalize a proper method of model calibration.

3. Many social simulation problems cannot be readily converted to mathematical

equations. This makes it difficult to apply some dynamic data assimilation techniques.

Therefore, for better long term predictions, we propose an adaptable framework, that would give the agent based micro-simulation (ABMS) the ability to consistently evolve with the target system, by continually absorbing the dynamically changing individual level and environmental level data, of this target system.

The objective is to create a framework that would consistently maintain the ABM's validity, while the behaviour of the environment or entities within the target system changes. To achieve this, we use an evolutionary algorithm approach[5]. This approach is used at the individual agent level, so that, as new data is absorbed by the agent, the agent adapts to the change occurring in the observed target system.

To do this, the simulation keeps track of the gap between the simulation result of each individual agent and the real data of its corresponding entities in the target system, and when the error becomes significant, it activates the agent's evolution process to reduce the error. The evolution process evolves the behavioural pattern of the agent, while calibrating its attributes, using incoming dynamic data. this allows a full impact of the changing data on the model. The result of this, is an adaptable model with consistent accuracy, in representing changing social systems.

6.1 Background

For an agent based micro simulation (ABMS) to be continuously accurate in policy analysis, the data used in creating the model has to be dynamically updated in order to ensure adaptability. To achieve such continuous updating, two difficulties have to be surmounted:

1. Establishing the decision making process of the agent within its modelled environment (this has been dealt with in chapter)

2. Adapting agent behaviours, attributes and model structure to a consistently changing target system.

This chapter aims to provide a novel framework that addresses the latter challenge such that the agent behavioural and attribute data at the individual level are dynamically updated, so that a consistently validated model would be used in policy analysis.

The challenge of dynamically calibrating consistently changing data into a model is generally referred to as dynamic data assimilation [104][57]. This has been commonly used in fields of geo-sciences, weather forecasting, hydrology and other physical and environmental systems. Many of the techniques used in achieving this include kalman filters [83], particle filters [31], reinforcement learning [20], genetic algorithm [77], 3D variation analysis [59] and 4D variation analysis [100].

To produce an ocean modelling system where the optimal estimation of the real ocean state is produced,[105] used the 4D variation algorithm in dynamically calibrating remotely sensed observation data from a space satellite into the simulating model. In a similar fashion, [9] used 3D variation to improve ozone simulations of a Mexico city basin so as to generate the optimal estimates of atmospheric states. Kalman filters have also been used extensively for dynamic data assimilation in ABMs. [104] Illustrates how ensemble kalman filters can be used for dynamic data assimilation in ABMs of complex social systems. This was demonstrated using an ABM of footfall counts in Leeds city metropolis, in which pedestrian-count real life footage was dynamically assimilated into the running simulation. Particle filters have also been used in creating dynamic data driven models by approximating the state of dynamic sys-

tems using particle weights and associated weights. [108] Proposed a framework for using particle filter in dynamic data-driven simulations with a case study of a wild fire simulation. Genetic algorithms (GA) have also been used in assimilating data into dynamic data-driven models [6][43]. New methodologies for reducing implementation times while using genetic algorithms for data assimilation was proposed in [24]. Reinforcement learning [95] has also been used in creating dynamic data-driven social agent based models. This was shown in [42] where the Korean housing market was simulated using a reinforcement algorithm to create a dynamic agent based model by using a hidden markov model (HMM) to estimate the states of the system during the reinforcement learning while the model validation was incrementally improved as new data was made available.

Many of the research methods for applying dynamic data assimilation methods in ABM have mainly used aggregated data as opposed to individual level data. Our aim is to use individual level dynamic data to calibrate our model such that each individual agent will constantly adapt and reflect any change in features or behaviour of its corresponding entity in the target system.

To consistently adapt a data-driven agent based model (DDABM) to a continuously changing social complex system, the model has to be calibrated with dynamic data. And in the case of an individual level model where the agents within the model are all unique, not only the attributes of the individual agents need to be calibrated but the individual behaviours of each agent and the model structure also. Therefore, in our aim to realize a self-evolving model, we propose a novel concept that promotes a dynamic re-configurable DDABM such that as the structure of the target system and the behaviour of its modelled entities change, this change will be captured by the model as the data continually collected from the target system is absorbed. This will allow

model flexibility and constant model adaptation.

It is expected that this method will broaden the scope of dynamic individual level data-driven models for social systems which have usually been conducted mainly through parameter tuning and aggregate data analysis.

6.2 Dynamic Data-Driven Agent Based Micro Simulation Framework

The dynamic data-driven (DDD) process is used in creating a self-evolving model through data, which continuously maintains a decreased error between the simulation and the social complex system as time increases [47]. Our framework enables individual agent's to self-evolve using an evolutionary approach that scores and stores the individual behaviour models of each agent in an individual repositories.

When change is detected in a target system's entity, and all the other stored models within its corresponding agent's repository do not match the new behaviour pattern, a new behaviour model is created from the immediate data collected within a time frame, from that entity. This model is then simulated and validated against forthcoming real data, after which this is stored for future use in the agent's model repository. Every new model is compared with the others in the agent's repository, and it either replaces a model or is discarded (if the capacity for storage is exhausted). Models that are used more frequently are scored higher.

To create a framework for an adaptable DDABMS from this approach, a three step process is introduced similar to that used in other literature [6] (as seen in the flow chart in figure6.1). A conceptual framework is shown in Figure6.2which shows how

the data moves within the framework.

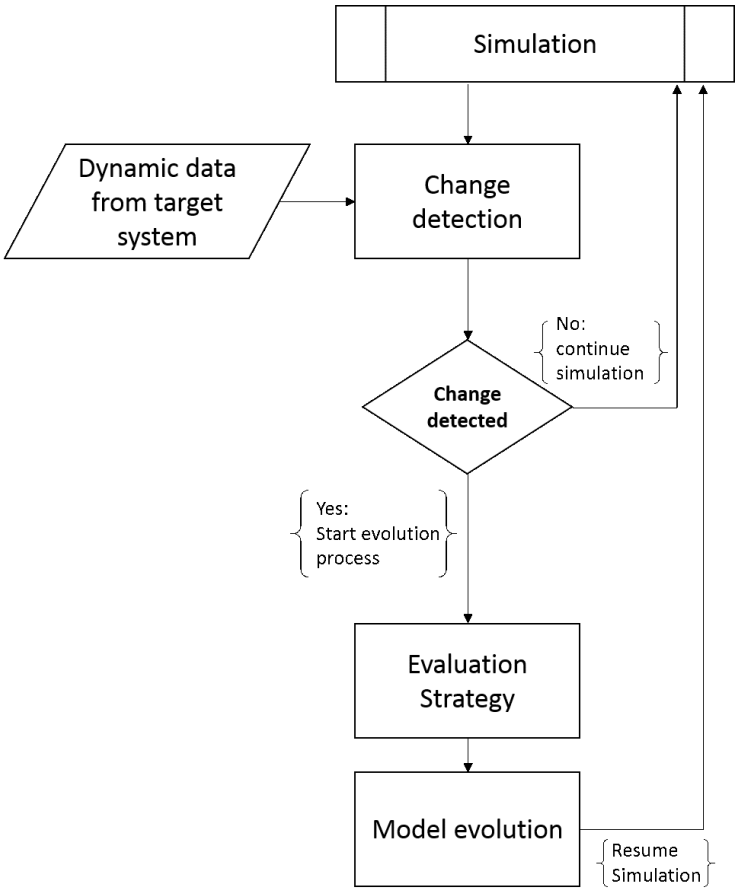

Figure 6.1: A simple flow chart describing the DDDABMS process

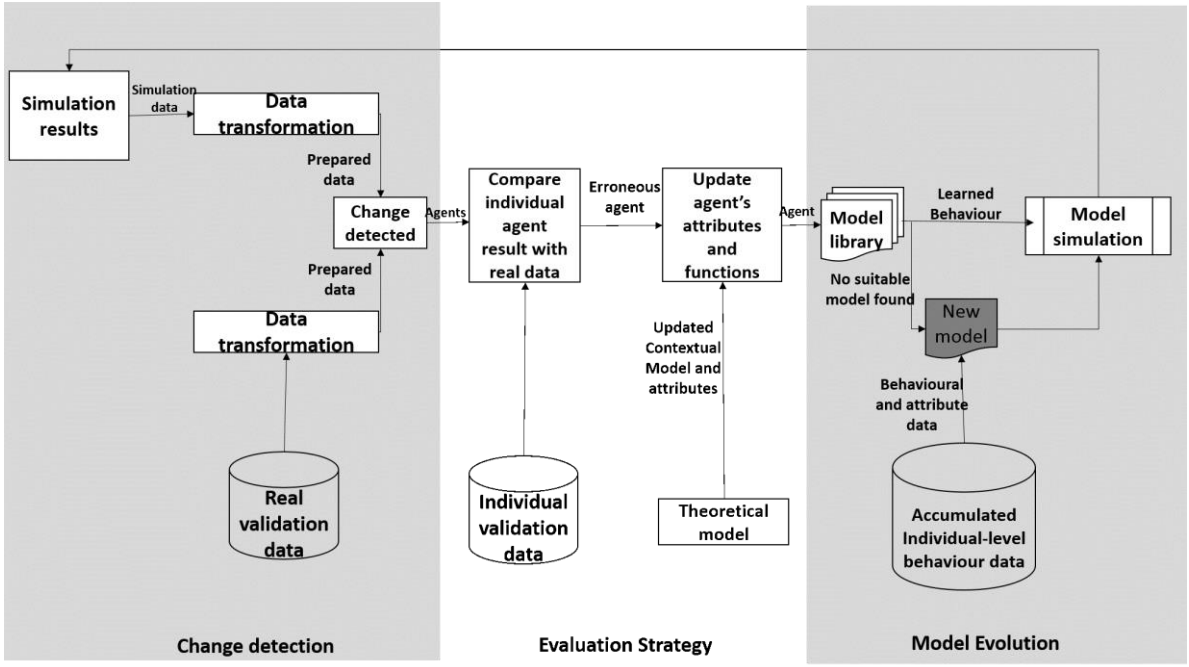

Figure 6.2: A conceptual data view of the framework

6.2.1 Change Detection

The change detection process determines if the emergent behaviour of the whole simulation has diverged from that of the target system. To ascertain such divergence, the real data from the target system has to be compared with the data from the simulation in a continuous fashion after every chosen time frame. For such comparison to be made, a first step of transformation must be taken so as to match the formation of the simulation data with that of the real data from the target system. This step may come with different challenges as the characteristics of the simulation data output such as its frequency and data unit is largely dependent on the ABMs objective and characteristic. Also, a variety of data from the target system may be used in comparing the simulated data. Therefore the aligning done in this step should be such that the final format of both data sets will enable quantitative analysis and comparison.

Ones that data has been transformed, change detection can then be conducted using statistical analysis of the difference between the real and simulated data. For instance, a point by point calculation can be done defining the average error at every time using statistical approaches like correlation coefficient test, root mean square errors [18] or mean absolute percentage error [25].

Depending on the kind of simulation, change can also be detected by focusing on the change in data trend, in which case model based estimation methods such as auto-regressive integrated moving average could be applied [38].

6.2.2 Evaluation Strategy

After the change detection process has established a deviation of the simulation from the actual target system, it activates an evaluation strategy process whose objective is to establish the cause of the deviation at the individual agent level. Its main objective is to isolate the erroneous agent and the cause of its error. This is done by engaging three different areas highlighted in literature for testing similar data-driven simulations [93][6][48]:

1. Individual attributes and environment parameters

2. Rule (or logic) based functions and sub-modules

3. Individual Behavioural models

For social systems, agent attributes and environment parameters are easily updated through observation, due to the fact that, they don't change as fast as physical systems [48]. Rule based functions, which could be code snippets (i.e, subroutines/functions/methods), localized at the agent level, or code snippets, globalized at the whole sim-

ulation level, can become outdated with time. Identifying this, will be based on expert knowledge, specific to the target system, which might be based on other experiments beyond the scope of this work. On the other hand, the actual behavioural pattern of the agents, can change unpredictably, which invariably renders the behavioural model of the agent annulled and the whole simulation erroneous. Therefore, the evaluation strategy module seeks to check the behavioural model of each of the agents, to isolate the erroneous agent with a behavioural pattern, different from that of its corresponding entity in the target system. Such agent, is then passed to the evaluation module for model adjustment using recent data.

6.2.3 Model Evolution

The model evolution process is the final stage of the self-evolving process using dynamic data. This happens at the individual agent level. It brings an adapting feature that goes beyond just updating parameters, but enables dynamic model reconfiguration. To do this, the evolution process requires an individual agent level model library\repository (such library can be created using different methods e.g. the suit approach in [109]), where behavioural model patterns are accumulated, scored and stored, overtime.

By having a library of n number of models within each agent's library, a survival of the fittest [66] evolutionary process is used in selecting the best model within the agent's library, for simulating the agent, per time. The algorithm is such that, the individual models (individual rules or pre-trained machine learning models), in the agent's model library, are used at different times, to model the agent during the life time of the simulation. As successive generations of the simulation pass, non-performing

models are eliminated from the agent's model library.

To select the right models within the agent's library par-time, a fitness characteristic is used to develop a scoring system that rates models. This rating system determines the position of a model within the agent's model library, and determines if a model should be added or discarded. An example of this is explained in section6.3, with scoring equations represented in Equation6.1to6.4.

The opportunity to populate an individual agent's model library, comes at every scheduled time lapse. At this time, the whole simulation is re-validated at global level, using real data. This is done at the change detection process stage. This re-validation will trigger the evaluation process stage, if a deviation is detected at the simulation's global level. The evaluation strategy process isolates the erroneous agent, which is then passed to the model evolution stage.In this stage all the the stored behavioural model in the agent's library will be evaluated against the real life validation data, to find a valid model. If none is found, the accumulated individual level data (since the last scheduled validation) of the specified entity is used in building a new behavioural pattern, which is then used in the agent simulation, after which, this model will then be be scored and stored in the repository. A finite number (i.e, an n number) of models are stored in the repository, and if that number is reached, the model with the least score is dropped in exchange the the new one.

As mentioned in the above paragraph, the whole simulation is re-validated at scheduled time lapses (e.g. every 3 days). At this point, the most accurate model is chosen from all the agents repositories. If non of the models in an agent's repository is accurate enough, the data stored within the scheduled time lapse, is used in creating a new model, which is also validated. If this model is also erroneous, more time would be given for data collection (i.e. the scheduled time lapse will be extended), before

using the stored behaviour data to create a new model. If this works, the timing for the scheduled time lapse should then be adjusted accordingly (i.e. it could be permanently changed from 3 days to 5 days). If this does not work, then the theoretical model used in creating the simulation (see Chapter 3) should be queried, as suggested in Chapter 3.

To summarise this whole process, a conceptual data view of the framework can be seen in figure6.2, showing how data moves within the whole framework. The change detection module compares the validation data with the real data to establish the presence of an error in the simulation, this then triggers the evaluation strategy which tries to establish the erroneous agent. When this is done, it further updates the isolated agent's attribute and rule based (logic only) functions, before passing it to the model evolution process, which then updates the agent's behavioural model. This updating is done either with a previously scored model in the library/repository or it uses an updated data from the corresponding entity in the target system to create a new behaviour model. After this, the whole simulation is then re-run with the corrected agent.

6.3 Distributed Network Behaviour Prediction Using the Dynamic Data-Driven Agent-Based Micro-Simulation Framework

To exemplify the dynamic data-driven agent based micro simulation (DDDABMS) framework explained in the previous section, we use the data-driven agent based network behaviour prediction model discussed in chapter5. Based on this, we aim to

make this micro simulation adaptable, such that in the event of a changing user behaviour (such as when a user's job specification changes, which invariably changes his usage pattern and application request), the model is robust enough to accommodate such behavioural change by adapting at an individual agent level to reflect the behavioural change of the corresponding user in the target system. This is achieved by consistently checking the simulation result against fresh data from the distributed network system (target system) and continuously keeping up to date individual agent model library/repositories so as to easily predict current user behaviour in the event of divergence in the simulation model's behaviour from that of the target system.

6.3.1 Distributed Network Behaviour Prediction

Distributed network behaviour is increasingly attracting huge attention both in academics and industrial initiatives, and most recently artificial intelligence has been used in leveraging its amazing power. Distributed networking is the network system over which computer programming, software, and its data are spread out across more than one computer but communicate complex messages through various dependent nodes.

Typically, a distributed network allows for the execution of distributed applications which results in complex behaviours among connected systems. This complexity in itself can create grey areas and vulnerabilities in the securities of these networks. Therefore, predicting the behaviour of these systems both at the macro and micro level has become essential.

In the past most researchers have predicted network behaviour and network attack patterns by using aggregated data, but in Chapter5we focused using the DDABMS model of chapter5and the application of ML at the individual user level, such that the

prediction of the individual network user behaviour pattern at the micro level becomes a substantive tool in creating a realistic agent based micro simulation of the whole distributed network. This in turn, served as a test-bed for exposing vulnerabilities within the target system.

In this chapter we will focus on updating the model consistently using dynamic data with the aim of making the model adaptable enough to continually reflect the network system while the behaviours of the individual users change with time. For example, if $User1$ at $workstation1$ (see chapter5) changes to a different job specification within the project, this will warrant a change in behaviour which will annul the behaviour of its corresponding agent in the simulation. This will create a ripple effect that will altogether change the behaviour of the target system and, by default, reduce the accuracy of the whole simulation.

In this section we intend to exemplify the proposed concept of the previous section by making the DDABM described in chapter5adaptable to behavioural and attribute changes occurring within the target system. This is achieved by passing in dynamic individual level behavioural and attribute data into the overall agent based model.

6.3.2 Change detection process

The change detection process determines if the emergent behaviour of the whole simulation has diverged from that of the target system. To achieve this, we continuously check the difference between the simulation request data on the agent server against the accumulated user request data on the real server. This is done after every 2 day time frame (this time frame is chosen based on expert advice specific to the project the network is being used for), during which the user's usage data will be collected

and stored. After every 2 days, if the correlation co-efficiency of the simulation result against the network server data goes below 0.8, or new attributes such as new applications have been added to the user's workstation, this will be flagged as a divergence of the simulation behaviour from that of the network, and would trigger the evaluation strategy process.

6.3.3 Evaluation strategy

The evaluation strategy in this experiment only concentrates on behavioural and attribute adaptation. It is a simple process used in determining which of the individual agent's result has deterred from its corresponding user's real data in the target system.

From Chapter5, we recall that the behavioural pattern was created from the generic chain of events (or activities) typically followed by all the users on the network, where: a user typically logs into a workstation at time t(where tET), and T is the total day time divided into time slots such that $T = [t_1, t_2, \ldots t_n]$ and n is the number of time slots), and makes a request R to the server for an application A_2 and A_3 while opening a local application A_1 (where A_1EA). Application A_1 access's a site with risk values A_{11}, A_{12} or A_{13} (Where A_{11}, A_{12} and A_{13} represent "Least Risky", "Moderately Risky" and "Very Risky" websites respectively). The event tuple can then be ordered as a chain of causal events which can be represented as a DAG: G(X, E) where $X = [x_1, x_2, \ldots x_n]$ is the chain of events with n being the number of events and x being each activity. E is the list of edges connecting each activity to its subsequent one as shown in figure 5.2 (see chapter5). This is then represented as a Bayesian network, such that the joint probability of all the nodes is a combination of the conditional probabilities of individual nodes. From this Bayesian network, we were able to create the behavioural

model of each network user. This individual models and the user attributes of each user the model was extracted from are then placed into individual agents which are then added to a virtual model representing the actual distributed network (i.e. the target system). This combination will create the data-driven agent based micro simulation (DDABMS) model of chapter5which will mimics the request made by actual users within the target network to actual network servers. In this evaluation strategy process, we track the accuracy of each user agent's model by checking the difference between the individual user agent's behaviour model simulation results against the real request of its corresponding users as recorded in the network server. This accuracy is based on a threshold derived from the Pearson correlation coefficient r [11][27], of the agent model against the real data of its corresponding user. See equation6.1.

$$agentaccuracy = \begin{cases} 1 \text{ if } r \geq 0.8 \\ 0 \text{ otherwise} \end{cases} \tag{6.1}$$

where $r=$

$$\frac{\sum_{i=1}^{n}(u_i a_i) - (\sum_{i=1}^{n} u_i)(\sum_{i=1}^{n} a_i)}{\sqrt{[\sum_{i=1}^{n} u_i^2 - (\sum_{i=1}^{n} u_i)^2][\sum_{i=1}^{n} a_i^2 - (\sum_{i=1}^{n} a_i)^2]}} \tag{6.2}$$

from Equation6.2, n is the number of time slots available for work on the network, which is equal to 3 ($i.e. t_1, t_2, and\ t_3$), u is the percentage of application request made by the user to the server at time slot i while a is the percentage of application request made by its corresponding agent to the server agent at the same time slot i. At the end of this process, any agent with an accuracy of 0 is filtered out for model evolution.

6.3.4 Model evolution

In the model evolution stage, we concentrate on the filtered agents and enable model reconfiguration using dynamic data. Prior to this, it should be noted that every agent in the simulation has a repository of models which have all been scored using a function S, which is a function of the correlation coefficient r of the model being scored and the immediate past score S_{n-1} of the model. It is also directly proportional to the number of times f this behaviour model was used after its last rejection

$$S = S_{n-1} + (S_0 * f) \tag{6.3}$$

and

$$S_0 = r \tag{6.4}$$

Once the model evolution process starts, the models in the repository are compared to the real data produced by the real user. This test is done starting from the model with the highest score within the repository. Once a model is found within the repository, that has $r \geq 0.8$ (when compared with the real data), this model is then passed into the user agent. If none is found, the data accumulated within the 2 days window prior to the change detection process is used in creating a new model which is then passed into the simulation and subsequently scored and stored in the agent's repository. A maximum of 3 models are stored in the agent's repository (this number is based on expert knowledge and storage capacity of the system memory in use). If a model is chosen from the repository, all the models are re-scored. This also happens if a new model is added to the repository, but in this case the least scored model could be discarded (Note: discarding happens if the models in the repository are up to three. This is done to make room for the new model). The new model is then added to the

repository while all the models are arranged according to their scores. As the agent's behaviour keeps changing, the model that represents his frequent behaviour pattern develops a high score and those with high scores are usually the first to be tested in the event of a pattern change in the user, thus allowing reduced time in reconfiguration.

6.3.5 Model Reconfiguration and Result Analysis

To evaluate the result, we focus on the data of $User3$ within the experiment. This user worked part time within the project and had different roles, therefore he consistently changed his schedule, causing a challenge in continually validating its corresponding agent's model. This started 5 days into the project, therefore rendering the model discussed in Chapter5inaccurate after 5 days. To solve this we used the adaptive method discussed in this paper and the process discussed in this section.

Table6.1below compares the validation results for $user3$'s corresponding agent as his schedule changed throughout the project. In this table we show the validation result using static data (which is the DDABM approach, where data is gathered at the beginning of the simulation and never updated) in creating models even as the behaviour of the user changes as oppose to using the adaptation process described in this section, which adapts as the user behaviour changes within the network. Table6.1 invariably compares the use of DDABMS to DDDABMS in our experiment, showing that if only static data is used in simulating the user behaviours in our network the simulation will only be accurate 50 percent of the time as opposed to over 90 percent of the time if a continuous data update and model adaptation approach is used. This can be visualized in the graph of Figure6.3

Project Day	Accuracy using	
	Static Data	*Dynamic Data*
Day3	0.89	0.89
Day5	0.89	0.89
Day7	0.36	0.9
Day9	0.89	0.89
Day11	-0.63	0.9
Day13	0.89	0.89
Day15	0.36	0.9
Day17	0.36	0.9

Table 6.1: Accuracy Comparison of One Agent

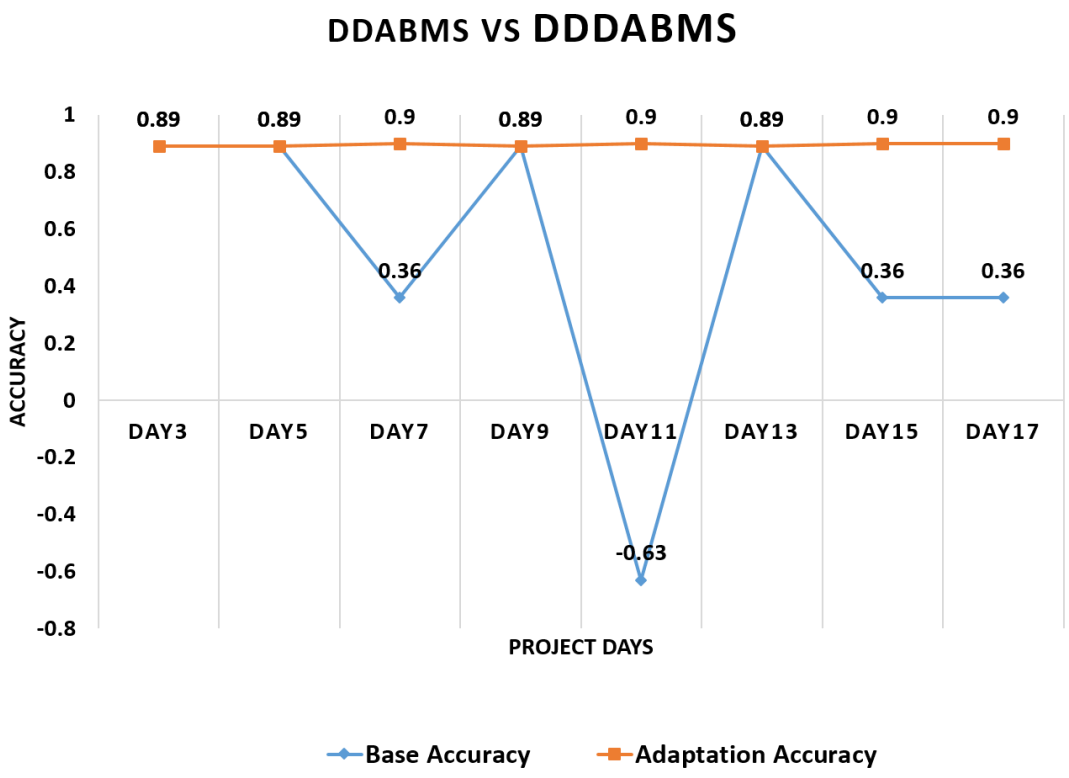

Figure 6.3: *agent*3's accuracy when using static data (using the DDABM) compared with when adapting with dynamic data (using the DDDABM).

On day 5, 7, 9 11 and 13, $user3$ changed his time schedule, thereby changing his behaviour pattern. This triggered the evaluation process in the next validation (i, 2 days later). A diary of this is explicitly written in table 6.2

Project Day	***Adaptation Diary***
Day3	First *Model*1 created from *user*3's day 1 and 2 behavioural and attribute data (i.e. model 1) and validated. *model*1's score is calculated using equation 6.3
Day5	*Model*1 validated with server data of day 4 and 5. Therefore *model*1 is still being used. *Model*1's new score was updated using equation6.3(*model*1's score increased since it was used again)
Day7	*User*3 changed schedule on day 5 therefore *model*1's validation (when compared to the server data) dropped below the accuracy threshold of 0.8. This prompted a new model (i.e. *model*2) to be created from Day 5 and 6's data. *Model*1 and *model*2's scores were recalculated
Day9	*User*3 changed schedule. Therefore *model*2's accuracy dropped below 0.8 when compared to the server data from day 7 till present, therefore it could not be used. When compared to *Model*1 the accuracy jumped back to 0.89. Therefore *Model*1 was used again. The scores for both models was updated.
Day11	*User*3 changed schedule. Comparing the server data of the last 2 days to that of the simulation result makes the accuracy of model 1 reduce below 0.8. This is the same when compared to *model*2. Therefore both models are bypassed and a new model (model 3) is created using day 9 and 10's data. *User*3 is scored while model 1 and 2's scores are updated
Day13	*User*3 changed schedule. Comparing the server data of the last 2 days to that of the simulation result makes the accuracy of *User*3 reduce below 0.8. This is the same when compared to *model*2. Therefore both models are bypassed but *model*1's accuracy meets the threshold and is then used. The 3 model's scores are updated.
Day15	*User*3 changed schedule again. Comparing the server data of the last 2 days to that of the simulation result makes the accuracy of model 1 reduce below 0.8. This is the same when compared to *User*3. Therefore both models are bypassed but *model*2's accuracy meets the threshold and is then used. The 3 model's scores are updated.
Day17	The present *model*2 being used Is revalidated and it still meets the required accuracy, therefore it remains in use and the adaptation module is not triggered.

Table 6.2: Diary of Model change in agent 3

Day	MODEL IN SIMU-LATION	MODEL 1 SCORE	MODEL 2 SCORE	MODEL 2 SCORE
Day3	Model 1	1.78	-	-
Day5	Model 1	3.56	-	-
Day7	Model 2	3.56	1.8	-
Day9	Model 1	4.45	1.8	-
Day11	Model 3	4.45	1.8	1.8
Day13	Model 1	5.34	1.8	1.8
Day15	Model 2	5.34	2.7	1.8
Day17	Model 2	5.34	4.5	1.8

Table 6.3: The score of each model used by agent 3 on each validation day

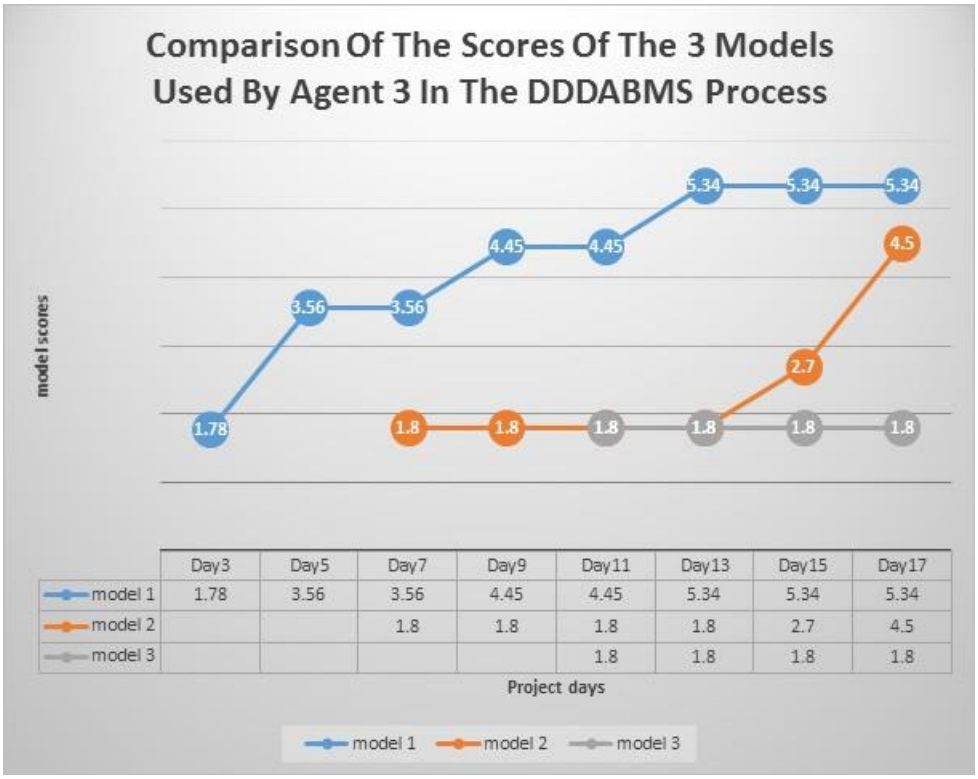

Figure 6.4: *agent*3's visualization of the adaptation of agent 3, showing the chronicle of the scoring of the 3 models used, starting from their formation day

Table6.3above chronologically gives the score of each of the models (using Equa-

tion6.3and6.4) used in adapting $agent3$ as $user3's$ schedule changed as the project days went by. This same table has been graphed for visualization in Figure6.4. It can be seen from Figure6.4that as the days went by model 1 was used more frequently, hence it developed higher scores.

6.4 Chapter Summary

Data-driven agent based micro simulation models used in predicting social complex systems have easily been rendered obsolete due to the consistent change in the behaviour of the entities within the system being modelled. This is due to the fact that the gathered data used in creating the model becomes obsolete when the behaviour of the entities being modelled changes. In this chapter, we addressed this issue by creating a framework for a dynamic data-driven agent based micro-simulation, which uses individual level consistently updated data (dynamic data) collected from the modelled entities within the social system to create agents.

This framework was applied to a previously simulated model of a distributed computer network. Using the framework, we were able to adapt this model in the face of changing network user behaviour, thereby maintaining the validity of the model and sustaining its accuracy. The next step in this project will be to apply big data tools in simulating this network model so as to accommodate as many network users as possible. This step will bring this project a step closer to industry application, as a decision making tool.

Chapter 7

Conclusions

In conclusion, this thesis has shown that, with the availability of individual level data, more accurate simulation models can now be created by representing each unique entity with a micro-agent created from data abstracted directly from the target system. Up until now, literature has not produced any structured approach which would allow both historic and dynamic individual level data, have full impact on the model, while creating unique agents directly from the target system's entities. The chapters in this thesis have proposed such approaches, while answering the relevant questions involved in the challenge.

With the proliferation of large scale behavioural data at micro/individual level, and the advancement of ML, IOT techniques and data mining tools, it has now become possible to create an agent based micro simulation (ABMS), whereby individual level empirical data can be used in generating agent behavioural rules, and in initializing agent attributes.

This approach has been used by some authors in creating models, but literature has not produced structured approaches to abstract agents directly from individual level

data , especially from dynamic data. Therefore, this research has aimed at developing novel methods, algorithms and and frameworks from data driven agent based micro simulation (DDABMS) that will focus on using static and dynamic individual level data to generate agent behavioural rules and initialize agent attributes and values.

This aim has been a result of the fact that most data driven models (DDM) have utilized aggregate data in producing their models, and most structured frameworks have accommodated only the use of aggregate data or neglected the fast growing presence of individual level data which can open the door to building unique models with every unique entity in the target social complex system accurately represented, thereby creating more accurate representations of the complex system.

In an attempt to bridge this research gap with the use of micro level data, we have merged the ABM technique with the MSM technique as suggested by [87][7][88].

Although these two techniques have the following short comings:

- Limited behaviour and agent interaction of MSM,
- Conceptual rule based focus in generating agent behaviour in ABM,
- Projection weakness of ABM,

we have shown that throughout this thesis, the combination of both techniques would produce ABMS that would:

- predict and analyse the transition links from local agent behaviours to global emergent outcomes. This would enable the possibility of predicting both the effect of a policy on an individual agent, and the effect of an individual agent on the policy.

- achieve consistency with the world outside a defined core system boundary, by merging empirical data with conceptual rules.

- simultaneously represent a target system's process on different spatial and temporal scales.

- integrate observable and conceptual behaviours, while retaining the ability of the model to achieve endogenous emergence.

This amalgamation is what allowed the ease in achieving the thesis objectives, which is to create structures and frameworks for a modelling approach that focuses on micro level static and dynamic data in generating an agent's behavioural rules and initializing an agent's attribute values. Such agents should then be used for the simulation of social complex systems.

Throughout this thesis, we have succeeded in answering the two research questions proposed in the introduction chapter1. The first research question involved the challenge of creating a novel framework for agent based micro-simulation, and the correct answer to his question must fulfil the four criteria enumerated in Section 1.4. Question one of this thesis was answered in Chapter 3, with the creation of the data driven agent based micro-simulation framework in chapter 3, section 3.1. The fist criteria to be fulfilled in answering this question, is that: *the proposed architecture should be generic enough to be applied in modelling any social complex system*. This was fulfilled in chapter 4 and 5 of this thesis. By successfully applying the proposed framework of chapter 3, to two different social complex systems, the generic nature of the proposed framework has been proven.

Also, the second and third criteria that must be fulfilled in correctly answering this first research question, was fulfilled in chapter 3. This was discussed in section 3.1,

where, process 3.4.2, 3.4.3, 3.4.4 and 3.4.5 of Figure 3.3, all show the process and data flow of how micro-level static data extracted from the smallest entity in the target system, can be used in attribute initialization and in the creation of behaviour rules of its corresponding agent. The framework then further showed how an overall simulation can be easily validated using data extracted from the target system.

The second research question, which involved the challenge of creating a novel framework that will fulfill all the criteria requested in question one, while also remaining adaptable, while the target system changes. This was answered in chapter 6, with the creation of the framework discussed in section 6.2. This adaptability was achieved by applying a model library approach at the individual agent level. This library of behaviour models, owned by individual agents, used an evolutionary algorithm in selecting the best behaviour model, for simulating the agent, par-time. This evolutionary algorithm, used a survival of the fittest approach in scoring the models within the agent's library, such that models that constantly fail at reflecting the agent's corresponding entity's behaviour within the target system, will be eliminated.

7.1 Research Contributions

In the process of answering the research questions, the following contributions where made, to the field of artificial intelligence:

1. A novel framework for agent based micro-simulations, with a focus on static behavioural data at the micro agent level, to create behavioural rules, and to initialize attributes, such that the overall simulation could be validated using empirical data.

2.A novel framework for agent based micro-simulations, with focus on dynamic behavioural data at the micro agent level, to create behavioural rules and to initialize attributes, such that the overall simulation could be validated using empirical data.

3.A novel test bed for testing the impact of customer reaction to price policy change in a public transport system.

4.A novel test bed for testing the vulnerability of a distributed computer network to a computer network virus attack.

In Chapter 3, we proposed a DDABMS approach using a three level top-down conceptual framework. The first level of the framework was a high level view of the elements involved in the modelling approach, while the population level view delved into the creation and parameterization of the agent population using a data driven approach. The micro/individual level view went deeper into establishing processes for directly infusing individual level static data to initialize the agent attribute and create behavioural rules for individual agents.

In this Chapter, was not just the first contribution of this thesis but also, in it lies the answer to the first research question. A Framework was indeed created which a generic architecture which concentrated on using a data driven concept which focused on the accurate representation of every individual in the target social system. The aim of this, as mentioned earlier, was to fill the gap of the absence of a methodological approach at the individual agent level in creating behavioural rules and initializing attributes from individual level static and historic data.

In Chapter 4, the first proof of concept for the proposed framework was established through the creation of a passenger rail model of the original Metro northern

rail (MNR) transport system in New York using the DDABMS approach discussed in chapter 3. This model took into cognisance every micro entity within the complex system, including every regular passenger, working train vehicle (including their physical characteristics, routes and schedules) and the names and spatial identities of all rail stations and rail lines used by MNR. This enabled the building of unique individual agents directly from data while extracting the individual target system's entity's behavioural pattern and attributes from the same data as required in the algorithm. Further more, empirical data from the the rail's system was then used in validating the model, thereby creating a novel test bed for which six PTO policies were tested.

In Chapter 5 we achieved the same aim of conceptualizing the proposed framework in Chapter 3, using a different social complex system. We created a model of a distributed computer network (DCN)using the DDABMS approach. Compared to Chapter 4, a process to process explanation approach was not used, but rather the focus was on the bottom-up characteristics of the framework, which was majorly done by representing the simulation flow with a flowchart.

Data mining and ML tools were used in extracting individual user behaviour from the user end of the network, which was then used in creating individual agents who made request for applications hosted in the server agent on the network. This sought to replicate the behaviour of the real life DCN, and the baseline simulation was validated by comparing temporal characteristics of the user agents request to that of the real user as recorded in the real network server. This baseline model then served as a test bed for predicting the vulnerability of the actual network to viral attack, thereby enabling easy risk policy creation by network administrators. From this process, we derived our fourth contribution.

Since the data driven ABMS approach discussed in Chapter 3 abstracts only his-

torical data from the target system, as time passes, error accumulates, thus negatively affecting the reliability of the results gotten from this model. This phenomenon is mainly due to the fact that human behaviours are subject to change with time, and the environment around the human subject can also change or instigate behavioural change, thereby annulling the attributes and behavioural patterns extracted from the historical data previously gathered. In other words,take for instance the passenger rail model simulated in Chapter 3, if after a few weeks the traveller's behaviour pattern changes, such that his route changes (this could be due to a change in his job location), this will invariably introduce a level of error that will impact the models accuracy, all due to the simulation's inability to ingest new data, hence being in-adaptable.

This challenge in itself ushers in the second research question detailed in Chapter 1 which is then answered in Chapter 6 by creating an adaptable framework for a dynamic data drive agent based micro simulation, which uses micro level consistently updated data (dynamic data) collected from the modelled entity within the target social system, to create agent models.

To prove this concept, we recreated the distributed network model discussed in Chapter 5, such that data was infused into the model throughout the course of the experiment. By adding an evolutionary module, the model was able to become adaptable as the individual level behavioural and attribute data of the users were updated, thereby keeping the accuracy of the model at above 80 percent as opposed to a 50 percent decline in accuracy without the adaptable approach.

7.2 Future Work

Obviously, the full scale implementation of this framework in representing full fledged target social systems will need the application of big data tools like map-reduce techniques and cloud processing infrastructure. Data storage tools like MongoDB will also be needed in storing micro level data after extraction. The next phase of this research will involve this aspect, so as to create more realistic models for policy effect prediction.

Also, the calibration of dynamic data into simulation has to be automated using existing software engineering processes. Such automation will involve the collection, storage and infusion of data into the simulation, with the aim of maintaining the models decision making accuracy through time.

One of the proposition in this thesis is that this framework is generic. Therefore, more research will be further done on applying this framework on other social systems. A particular area that will fit in, is in the simulation of social media chats, since historical blogs of individual users can be extracted and mined. This will allow the possibility of mining the behaviour pattern of individual platform users. Such patterns can be used in modelling the individual user behaviour. If this process is followed up on a number of associated users within a social media platform, the emergent behaviour of the whole platform can be recreated by simulation. This would allow the exploration of different interesting ”what-if” scenarios, that would usually not be tested on the actual platform.

In conclusion, this thesis has shown that with the infiltration of individual level data, more accurate simulation models can now be created by representing each unique entity within the social system with micro agents created from data abstracted directly

from their corresponding entity. Up until now, literature has not produced any structured approach which would allow both historic and dynamic individual level data have full impact on the model while creating unique agents directly from the target systems entities. The Chapters in this thesis has proposed such models, while answering the relevant questions involved in the challenge.

References

[1]D. Agarwal, J. M. Gonzalez, G. Jin, and B. Tierney, “An infrastructure for passive network monitoring of application data streams,” 2003.

[2]M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, “Machine learning in wireless sensor networks: Algorithms, strategies, and applications,” *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014.

[3]V. Ankam, *Big data analytics*. Packt Publishing Ltd, 2016.

[4]R. Arroyo-Valles, R. Alaiz-Rodriguez, A. Guerrero-Curieses, and J. Cid-Sueiro, “Q-probabilistic routing in wireless sensor networks,” in *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*. IEEE, 2007, pp. 1–6.

[5]T. B äck, D. B. Fogel, and Z. Michalewicz, *Evolutionary computation 1: Basic algorithms and operators*. CRC press, 2018.

[6]J. W. Bae, E. Paik, D.-o. Kang, J. Jung, and C.-H. Lee, “Simulation framework for self-evolving agent-based models: a case study of housing market model,”

in *Proceedings of the 2018 Winter Simulation Conference*. IEEE Press, 2018, pp. 1120–1131.

[7]J. W. Bae, E. Paik, K. Kim, K. Singh, and M. Sajjad, "Combining microsimulation and agent-based model for micro-level population dynamics," *Procedia Computer Science*, vol. 80, pp. 507–517, 2016.

[8]A. Bassolas, J. J. Ramasco, R. Herranz, and O. G. Cant ú-Ros, "Mobile phone records to feed activity-based travel demand models: Matsim for studying a cordon toll policy in barcelona," *Transportation Research Part A: Policy and Practice*, vol. 121, pp. 56–74, 2019.

[9]N. Bei, B. de Foy, W. Lei, M. Zavala, and L. Molina, "Using 3dvar data assimilation system to improve ozone simulations in the mexico city basin." *Atmospheric Chemistry and Physics*, vol. 8, no. 24, 2008.

[10]A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *arXiv preprint arXiv:1306.6709*, 2013.

[11]J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*. Springer, 2009, pp. 1–4.

[12]M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 3, pp. 1136–1159, 2012.

[13]R. Boero and F. Squazzoni, "Does empirical embeddedness matter? methodological issues on agent-based models for analytical social science," *Journal of artificial societies and social simulation*, vol. 8, no. 4, 2005.

[14]E. Bruch and J. Atwell, "Agent-based models in empirical social research," *Sociological methods and research*, vol. 44, no. 2, pp. 186–221, 2015.

[15]A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[16]B. A. Butrica, R. W. Johnson, and K. E. Smith, "Potential impacts of the great recession on future retirement incomes," *Reshaping retirement security: lessons from the global financial crisis*, pp. 36–63, 2012.

[17]A. Caiani, A. Godin, E. Caverzasi, M. Gallegati, S. Kinsella, and J. E. Stiglitz, "Agent based-stock flow consistent macroeconomics: Towards a benchmark model," *Journal of Economic Dynamics and Control*, vol. 69, pp. 375–408, 2016.

[18]T. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature," *Geoscientific model development*, vol. 7, no. 3, pp. 1247–1250, 2014.

[19]J. Chen, E. Haber, R. Kang, G. Hsieh, and J. Mahmud, "Making use of derived personality: The case of social media ad targeting," in *Ninth International AAAI Conference on Web and Social Media*, 2015.

[20]Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 285–292.

[21]S. H. Collado, "Towards a data-driven approach for agent-based modelling: simulating spanish postmodernisation," *Universidad Complutense de Madrid, Madrid*, 2009.

[22]J. Conway, "The game of life," *Scientific American*, vol. 223, no. 4, p. 4, 1970.

[23]M. H. Davis, *Markov models and optimization*. Routledge, 2018.

[24]D. De, W.-Z. Song, M. Xu, C.-L. Wang, D. Cook, and X. Huo, "Findinghumo: Real-time tracking of motion trajectories from anonymous binary sensing in smart environments," in *2012 IEEE 32nd International Conference on Distributed Computing Systems*. IEEE, 2012, pp. 163–172.

[25]A. De Myttenaere, B. Golden, B. Le Grand, and F. Rossi, "Mean absolute percentage error for regression models," *Neurocomputing*, vol. 192, pp. 38–48, 2016.

[26]S. Dua and X. Du, *Data mining and machine learning in cybersecurity*. Auerbach Publications, 2016.

[27]R. W. Emerson, "Causation and pearson's correlation coefficient," *Journal of visual impairment and blindness*, vol. 109, no. 3, pp. 242–244, 2015.

[28]Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2432–2455, 2017.

[29]M. M. Favreault, R. W. Johnson, K. E. Smith, and S. R. Zedlewski, "Boomers'

retirement income prospects," *Urban Institute Program on Retirement Policy, Brief*, vol. 34, 2012.

[30]M. M. Favreault, K. E. Smith, and R. W. Johnson, "The dynamic simulation of income model (dynasim)," *Urban Institute Program on Retirement Policy Research Report*, 2015.

[31]P. Fearnhead and H. R. K ünsch, "Particle filters and data assimilation," 2018.

[32]E. Frydenlund, "Dynamic attitudes about motherhood: Modeling family planning policies in japan," 2011.

[33]M. Gatti, P. Cavalin, S. B. Neto, C. Pinhanez, C. dos Santos, D. Gribel, and A. P. Appel, "Large-scale multi-agent-based modeling and simulation of microblogging-based online social network," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*. Springer, 2013, pp. 17–33.

[34]N. Gilbert, *Agent-based models*. Sage, 2008, no. 153.

[35]V. Grover, R. H. Chiang, T.-P. Liang, and D. Zhang, "Creating strategic business value from big data analytics: A research framework," *Journal of Management Information Systems*, vol. 35, no. 2, pp. 388–423, 2018.

[36]R. Heijungs and J. B. Guine, "Allocation and 'what-if'scenarios in life cycle assessment of waste management systems," *Waste management*, vol. 27, no. 8, pp. 997–1005, 2007.

[37]A. Horni, K. Nagel, and K. W. Axhausen, *The multi-agent transport simulation MATSim*. Ubiquity Press London, 2016.

[38]E. A. Jackson, "Comparison between static and dynamic forecast in autoregressive integrated moving average for seasonally adjusted headline consumer price index," *Available at SSRN 3162606*, 2018.

[39]T. Jensen and E. J. Chappin, "Automating agent-based modeling: Data-driven generation and application of innovation diffusion models," *Environmental modelling & software*, vol. 92, pp. 261–268, 2017.

[40]G. Jovicic, "Activity based travel demand modelling," *Danmarks Transp. Skn*, 2001.

[41]K. N. Junejo and J. Goh, "Behaviour-based attack detection and classification in cyber physical systems using machine learning," in *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*. ACM,2016, pp. 34–43.

[42]D.-o. Kang, J. W. Bae, C. Lee, J.-Y. Jung, and E. Paik, "Data assimilation technique for social agent-based simulation by using reinforcement learning," in *Proceedings of the 22nd International Symposium on Distributed Simulation and Real Time Applications*. IEEE Press, 2018, pp. 220–221.

[43]D.-O. Kang, J. W. Bae, and E. Paik, "Incremental self-evolving framework for agent-based simulation," in *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2016, pp. 1428–1429.

[44]H. Kavak, J. J. Padilla, C. J. Lynch, and S. Y. Diallo, "Big data, agents, and machine learning: towards a data-driven agent-based modeling approach," in *Proceedings of the Annual Simulation Symposium*. Society for Computer Simulation International, 2018, p. 12.

[45]H. Kavak, D. Vernon-Bido, and J. J. Padilla, "Fine-scale prediction of people's home location using social media footprints," in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 2018, pp. 183–189.

[46]C. Kennedy and G. Theodoropoulos, "Adaptive intelligent modelling for the social sciences: Towards a software architecture," *SCHOOL OF COMPUTER SCIENCE RESEARCH REPORTS-UNIVERSITY OF BIRMINGHAM CSR*, vol. 11, 2006.

[47]C. Kennedy, G. Theodoropoulos, V. Sorge, E. Ferrari, P. Lee, and C. Skelcher, "Aimss: An architecture for data driven simulations in the social sciences," in *International Conference on Computational Science*. Springer, 2007, pp. 1098–1105.

[48]C. Kennedy, G. Theodoropoulos, V. Sorge, E. Ferrari, P. Lee, and C. Skelcher, "Data driven simulation to support model building in the social sciences," *Journal of Algorithms and Computational Technology*, vol. 5, no. 4, pp. 561–581, 2011.

[49]W. G. Kennedy, "Modelling human behaviour in agent-based models," in *Agent-based models of geographical systems*. Springer, 2012, pp. 167–179.

[50]E. Khademi and H. Timmermans, "Incorporating traveler response to pricing policies in comprehensive activity-based models of transport demand, literature review and conceptualisation," *Procedia-Social and Behavioral Sciences*, vol. 20, pp. 594–603, 2011.

[51]B. Kickhfer, D. Hosse, K. Turner, and A. Tirachini, "Creating an open matsim scenario from open data: The case of santiago de chile," *http://www. vsp. tu-berline. de/publication: TU Berlin, Transport System Planning and Transport Telematics*, 2016.

[52]P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[53]T. A. Kugler and C. A. Fitch, "Interoperable and accessible census and survey data from ipums," *Scientific data*, vol. 5, p. 180007, 2018.

[54]J. Kwapie and S. Drod, "Physical approach to complex systems," *Physics Reports*, vol. 515, no. 3-4, pp. 115–226, 2012.

[55]D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, "A survey of deep learning-based network anomaly detection," *Cluster Computing*, pp. 1–13, 2017.

[56]C. T. Lawson, "Applying census data for transportation: 50 years of transportation planning data progress," *Transportation Research Circular*, no. E-C233, 2018.

[57]J. M. Lewis, S. Lakshmivarahan, and S. Dhall, *Dynamic data assimilation: a least squares approach*. Cambridge University Press, 2006, vol. 13.

[58]J. Lismont, T. Van Calster, M. Óskarsdóttir, S. Vanden Broucke, B. Baesens, W. Lemahieu, and J. Vanthienen, "Closing the gap between experts and novices

using analytics-as-a-service: an experimental study," *Business and Information Systems Engineering*, pp. 1–15, 2018.

[59]C. Liu, X. Shao, and W. Li, "Multi-sensor observation fusion scheme based on 3d variational assimilation for landslide monitoring," *Geomatics, Natural Hazards and Risk*, vol. 10, no. 1, pp. 151–167, 2019.

[60]M. Lovri ć, T. Li, and P. Vervest, "Sustainable revenue management: A smart card enabled agent-based modeling approach," *Decision Support Systems*, vol. 54, no. 4, pp. 1587–1601, 2013.

[61]G. R. Madey, A.-L. Barab ási, N. V. Chawla, M. Gonzalez, D. Hachen, B. Lantz, A. Pawling, T. Schoenharl, G. Szabó, P. Wang *et al.*, "Enhanced situational awareness: Application of dddas concepts to emergency and disaster management," in *International Conference on Computational Science*. Springer, 2007, pp. 1090–1097.

[62]B. Mahdavi, D. O'Sullivan, and P. Davis, "An agent-based microsimulation framework for investigating residential segregation using census data," 2007.

[63]Makinde. (2017 (accessed: 08.02.2018)) Agent based micro-simulation of a passenger rail system.https://github.com/lolumak.

[64]O. Makinde, D. Neagu, and M. Gheorghe, "Agent based micro-simulation of a passenger rail system using customer survey data and an activity based approach," in *UK Workshop on Computational Intelligence*. Springer, 2018, pp. 123–137.

[65]J. Mandel, J. D. Beezley, A. K. Kochanski, V. Y. Kondratenko, and M. Kim,

"Assimilation of perimeter data and coupling with fuel moisture in a wildland fire–atmosphere dddas," *Procedia Computer Science*, vol. 9, pp. 1100–1109, 2012.

[66]D. Mayer, J. Belward, H. Widell, and K. Burrage, "Survival of the fittest—genetic algorithms versus evolution strategies in the optimization of sys- tems models," *Agricultural Systems*, vol. 60, no. 2, pp. 113–122, 1999.

[67]F. D. McKenzie, "Systems modeling: analysis and operations research," *Modeling and simulation fundamentals: Theoretical underpinnings and practical domains*, pp. 147–180, 2010.

[68]M. G. McNally and C. R. Rindt, "The activity-based approach," in *Handbook of Transport Modelling: 2nd Edition*. Emerald Group Publishing Limited, 2007, pp. 55–73.

[69]A. Mislove, S. Lehmann, Y.-Y. Ahn, J.-P. Onnela, and J. N. Rosenquist, "Understanding the demographics of twitter users," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[70]A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1. ACM, 2005, pp. 50–60.

[71]R. J. Morrison, "Making pensions out of nothing at all," in *International Microsimulation Conference on Population, Ageing and Health, NATSEM, University of Canberra, Australia*, 2003, pp. 7–12.

[72]S. Moss and B. Edmonds, "Sociology and simulation: Statistical and qualitative

cross-validation," *American journal of sociology*, vol. 110, no. 4, pp. 1095–1131, 2005.

[73]K. V. Mulcahy, "Cultural policy: Definitions and theoretical approaches," *The journal of arts management, law, and society*, vol. 35, no. 4, pp. 319–330, 2006.

[74]N. J. Nagelkerke *et al.*, "A note on a general definition of the coefficient of determination," *Biometrika*, vol. 78, no. 3, pp. 691–692, 1991.

[75]T. T. Nguyen and G. J. Armitage, "A survey of techniques for internet traffic classification using machine learning." *IEEE Communications Surveys and Tutorials*, vol. 10, no. 1-4, pp. 56–76, 2008.

[76]J. Nowak, M. Korytkowski, and R. Scherer, "Classification of computer network users with convolutional neural networks," in *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2018, pp. 501–504.

[77]F. Oloo and G. Wallentin, "An adaptive agent-based model of homing pigeons: A genetic algorithm approach," *ISPRS International Journal of Geo-Information*, vol. 6, no. 1, p. 27, 2017.

[78]W. V. Oortmerssen. Windows time tracking application. [Online]. Available: https://github.com/aardappel/procrastitracker

[79]OpenStreetMap.com. (2001 (accessed: 29.07.2019)) Metro-north rail line. https://www.openstreetmap.org/map=5/51.500/-0.100.

[80]G. H. Orcutt, "A new type of socio-economic system," *The review of economics and statistics*, pp. 116–123, 1957.

[81]S. Rasouli and H. Timmermans, "Activity-based models of travel demand promises, progress and prospects," *International Journal of Urban Sciences*, vol. 18, no. 1, pp. 31–60, 2014.

[82]I. Resource Systems Group. (2007 (accessed: 29.07.2019)) Metro-north origin-destination survey.http://web.mta.info/mta/planning/data/MTA_MNR-Survey-Final-Report.pdf.

[83]M. Roth, "Advanced kalman filtering approaches to bayesian state estimation," Ph.D. dissertation, Linköping University Electronic Press, 2017.

[84]R. K. Sawyer and R. K. S. Sawyer, *Social emergence: Societies as complex systems*. Cambridge University Press, 2005.

[85]T. C. Schelling, "Dynamic models of segregation," *Journal of mathematical sociology*, vol. 1, no. 2, pp. 143–186, 1971.

[86]K. Shantz and L. E. Fox, "Precision in measurement: Using state-level supplemental nutrition assistance program and temporary assistance for needy families administrative records and the transfer income model (trim3) to evaluate poverty measurement," *Washington, DC: US Census Bureau*, 2018.

[87]K. Singh, C.-W. Ahn, E. Paik, J. W. Bae, and C.-H. Lee, "A micro-level data-calibrated agent-based model, the synergy between microsimulation and agent-based modeling," *Artificial life*, vol. 24, no. 02, pp. 128–148, 2018.

[88]K. Singh, M. Sajjad, and C.-W. Ahn, "Towards full scale population dynamics modelling with an agent based and micro-simulation based framework," in

2015 17th International Conference on Advanced Communication Technology (ICACT). IEEE, 2015, pp. 495–501.

[89]A. Sirbu and O. Babaoglu, "Towards data-driven autonomics in data centers," in *2015 International Conference on Cloud and Autonomic Computing*. IEEE, 2015, pp. 45–56.

[90]A. Smajgl, D. G. Brown, D. Valbuena, and M. G. Huigen, "Empirical characterisation of agent behaviours in socio-ecological systems," *Environmental Modelling and Software*, vol. 26, no. 7, pp. 837–844, 2011.

[91]K. E. Smith and E. J. Toder, "Adding employer contributions to health insurance to social security's earnings and tax base," *Boston College Center for Retirement Research Working Paper*, no. 2014-3, 2014.

[92]R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE symposium on security and privacy*. IEEE, 2010, pp. 305–316.

[93]J. Song, B. Xiang, X. Wang, L. Wu, and C. Chang, "Application of dynamic data driven application system in environmental science," *Environmental Reviews*, vol. 22, no. 3, pp. 287–297, 2014.

[94]S.-C. Su, "Summarized network behavior prediction," *arXiv preprint arXiv:1705.01143*, 2017.

[95]R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[96]M. Svetlik, M. Leonetti, J. Sinapov, R. Shah, N. Walker, and P. Stone, "Automatic curriculum graph generation for reinforcement learning agents," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[97]R. K. Thomas, "Data sources for demography," in *Concepts, Methods and Practical Applications in Applied Demography*. Springer, 2018, pp. 31–51.

[98]P. M. Todd, F. C. Billari, and J. Sim ão, "Aggregate age-at-marriage patterns from individual mate-search heuristics," *Demography*, vol. 42, no. 3, pp. 559–574, 2005.

[99]T. M. Truong, F. Amblard, B. Gaudou, and C. S. Blanc, "Cfbm-a framework for data driven approach in agent-based modeling and simulation," in *International Conference on Nature of Computation and Communication*. Springer, 2016, pp. 264–275.

[100]N. Usui, Y. Fujii, K. Sakamoto, and M. Kamachi, "Development of a four-dimensional variational assimilation system for coastal data assimilation around japan," *Monthly Weather Review*, vol. 143, no. 10, pp. 3874–3892, 2015.

[101]P. Van Liedekerke, M. Palm, N. Jagiella, and D. Drasdo, "Simulating tissue mechanics with agent-based models: concepts, perspectives and some novel results," *Computational particle mechanics*, vol. 2, no. 4, pp. 401–444, 2015.

[102]M. van Steen and A. S. Tanenbaum, "A brief introduction to distributed systems," *Computing*, vol. 98, no. 10, pp. 967–1009, 2016.

[103]A. Walker, S. Fischer, and R. Percival, *A microsimulation model of Australia's*

Pharmaceutical Benefits Scheme. National Centre for Social and Economic Modelling, 1998.

[104]J. A. Ward, A. J. Evans, and N. S. Malleson, "Dynamic calibration of agent-based models using data assimilation," *Royal Society open science*, vol. 3, no. 4, p. 150703, 2016.

[105]J. Wilkin, J. Zavala-Garay, J. Levin, and W. G. Zhang, "Four-dimensional variational assimilation of satellite temperature and sea level data in the coastal ocean and adjacent deep sea," in *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, vol. 3. IEEE, 2008, pp. III–427.

[106]N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 5–16, 2006.

[107]C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.

[108]H. Xue, F. Gu, and X. Hu, "Data assimilation using sequential monte carlo methods in wildfire spread simulation," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 22, no. 4, p. 23, 2012.

[109]B. P. Zeigler, C. Seo, and D. Kim, "System entity structures for suites of simulation models," *International Journal of Modeling, Simulation, and Scientific Computing*, vol. 4, no. 03, p. 1340006, 2013.

[110]H. Zhang, Y. Vorobeychik, J. Letchford, and K. Lakkaraju, "Data-driven agent-based modeling, with application to rooftop solar adoption," *Autonomous Agents and Multi-Agent Systems*, vol. 30, no. 6, pp. 1023–1049, 2016.

Appendix A

Code

```
1  import java.io.BufferedReader;
2  import java.io.FileReader;
3  import java.io.IOException;
4  import java.util.HashMap;
5  import java.util.Map;
6  import java.util.Random;
7  import org.matsim.api.core.v01.Coord;
8  import org.matsim.api.core.v01.Id;
9  import org.matsim.api.core.v01.Scenario;
10 import org.matsim.api.core.v01.network.Network;
11 import org.matsim.api.core.v01.population.Activity;
12 import org.matsim.api.core.v01.population.Leg;
13 import org.matsim.api.core.v01.population.Person;
14 import org.matsim.api.core.v01.population.Plan;
15 import org.matsim.api.core.v01.population.PopulationWriter;
16 import org.matsim.core.config.ConfigUtils;
17 import org.matsim.core.gbl.MatsimRandom;
18 import org.matsim.core.network.io.MatsimNetworkReader;
19 import org.matsim.core.scenario.ScenarioUtils;
20 import org.matsim.core.utils.geometry.CoordUtils;
21 import org.matsim.core.utils.geometry.CoordinateTransformation;
22 import org.matsim.core.utils.geometry.geotools.MGC;
23 import org.matsim.core.utils.geometry.transformations.GeotoolsTransformation;
```

```
import org.matsim.core.utils.gis.ShapeFileReader;
import org.matsim.core.utils.io.tabularFileParser.TabularFileHandler;
import org.matsim.core.utils.io.tabularFileParser.TabularFileParser;
import org.matsim.core.utils.io.tabularFileParser.TabularFileParserConfig;
import org.opengis.feature.simple.SimpleFeature;
import com.vividsolutions.jts.geom.Geometry;
import com.vividsolutions.jts.geom.GeometryFactory;
import com.vividsolutions.jts.geom.Point;
import com.vividsolutions.jts.io.ParseException;
import com.vividsolutions.jts.io.WKTReader;

//This method is used in creating the passenger agent demand
public class CreateDemand2 {

  private static final String NETWORKFILE
          = "modeled_rail_line_with_stops.xml";
  private static final String KREISE
          = "new-york_osm_buildings.shp";

  private static final String PLANSFILEOUTPUT = "20TESTPLAN.xml";

  private static final String pusTripsFile
          = "random1%_of_travelSurvey_trips.txt";
  private static final String SHOPS = "shops.txt";
  private Scenario scenario;
  private Map<String,Geometry> shapeMap;
  private static double SCALEFACTOR = 0.1;
  private Map<String,Coord> Home;
  private Map<String,Coord> Work;
  private Map<String,Coord> shops;


  private Map<String,Coord> HomefromGeometry = new HashMap<>();
  private Map<String,Coord> WorkfromGeometry = new HashMap<>();


  CreateDemand2 (){
```

```
61      this.scenario = ScenarioUtils.createScenario(ConfigUtils.createConfig());
62      Network net = scenario.getNetwork();
63      MatsimNetworkReader onr = new MatsimNetworkReader(net);
64      onr.readFile(NETWORKFILE);
65    }
66
67    //This is the main method
68    public static void main(String[] args) {
69
70      CreateDemand2 cd = new CreateDemand2();
71      cd.run();
72    }
73
74    private void run() {
75      this.shapeMap = readShapeFile(KREISE, "type");
76
77      //CB-CB
78      double commuters = 200*SCALEFACTOR;
79      double carcommuters = 0.55 * commuters;
80
81      System.out.println(this.shapeMap.keySet());
82
83
84      //Create environment
85      for(String key:this.shapeMap.keySet()){
86         if (key.equals("apartments") ||
87         key.equals("apartment") ||
88         key.equals("residential") ||
89         key.equals("house")){
90
91           Geometry home = this.shapeMap.get(key);
92           Coord homecoordinate = drawRandomPointFromGeometry(home);
93           this.HomefromGeometry.put(key, homecoordinate);
94
95         }
96
97         if (key.equals("commercial") ||
```

```
98          key.equals("industrial") ||
99          key.equals("office") ||
100         key.equals("comapny") ||
101         key.equals("brewery")){
102
103          Geometry work = this.shapeMap.get(key);
104          Coord workcoordinate = drawRandomPointFromGeometry(work);
105          this.WorkfromGeometry.put("work", workcoordinate);
106       }
107   }
108   System.out.println(this.shapeMap.size()+"␣"
109           +this.shapeMap.get("commercial") );
110   System.out.println(this.HomefromGeometry.keySet()
111           +"␣"+this.HomefromGeometry.size());
112
113
114   //Crete passenger agent
115
116     try{
117       BufferedReader bufferedReader
118          = new BufferedReader(new FileReader(pusTripsFile));
119       bufferedReader.readLine(); //skip header
120
121       int index_personId = 0;
122       int index_xCoordOrigin = 11;
123       int index_yCoordOrigin = 10;
124       int index_xCoordDestination = 13;
125       int index_yCoordDestination = 12;
126       int index_activityDuration = 8;
127       int index_mode = 14;
128       int index_activityType = 2;
129
130       String line;
131     while((line = bufferedReader.readLine()) != null){
132       String parts[] = line.split(",");
133
134
```

```
135         int i = 0;
136         String mode = "car";
137         mode = "pt"; //if (i>carcommuters) mode = "pt";
138
139         //Get agents activity points
140         Coord homepoint
141             = createPoint(Double.parseDouble(parts[index_xCoordOrigin]),
142                 Double.parseDouble(parts[index_yCoordOrigin]));
143                   this.kindergartens)+"closest␣shop:"+t
144                   his.findClosestCoordFromMap(homec, this.shops));
145
146         Coord workpoint
147             = createPoint(Double.parseDouble(parts[index_xCoordDestination]),
148                           Double.parseDouble(parts[index_yCoordDestination]));
149
150         Coord closestHomepoint
151             = this.findClosestCoordFromMap(homepoint, HomefromGeometry);
152         Coord closestWorkpoint
153             = this.findClosestCoordFromMap(workpoint, WorkfromGeometry);
154
155         //find random points closest to home and work
156         System.out.println("id␣"+ parts[index_personId]+"␣" +
157         parts[index_xCoordOrigin]+"␣" +
158         parts[index_yCoordOrigin]+"␣Home␣point␣"+
159         closestHomepoint);
160
161
162         //create passenger
163         createOnePerson(Integer.parseInt(parts[index_personId])
164             , closestHomepoint, closestWorkpoint, mode);
165
166         }
167       bufferedReader.close();
168
169       }catch (IOException e) {
170           e.printStackTrace();
171           }
```

```
172     //}
173
174     PopulationWriter pw
175         = new PopulationWriter(scenario.getPopulation(),scenario.getNetwork());
176     pw.write(PLANSFILEOUTPUT);
177   }
178
179
180   private void createOnePerson(int i, Coord coord,
181       Coord coordWork, String mode, String toFromPrefix) {
182
183     Id<Person> personId = Id.createPersonId(toFromPrefix+i);
184     Person person = scenario.getPopulation().getFactory().createPerson(personId);
185
186     Plan plan = scenario.getPopulation().getFactory().createPlan();
187
188
189     Activity home
190       = scenario.getPopulation().getFactory().createActivityFromCoord("home", coord);
191     home.setEndTime(9*60*60);
192     plan.addActivity(home);
193
194     Leg hinweg = scenario.getPopulation().getFactory().createLeg(mode);
195     plan.addLeg(hinweg);
196
197     Activity work
198       = scenario.getPopulation().getFactory()
199             .createActivityFromCoord("work", coordWork);
200     work.setEndTime(17*60*60);
201     plan.addActivity(work);
202
203     Leg rueckweg = scenario.getPopulation().getFactory().createLeg(mode);
204     plan.addLeg(rueckweg);
205
206     Activity home2
207       = scenario.getPopulation().getFactory().createActivityFromCoord("home", coord);
208     plan.addActivity(home2);
```

```
209
210     person.addPlan(plan);
211     scenario.getPopulation().addPerson(person);
212   }
213
214   private Coord drawRandomPointFromGeometry(Geometry g) {
215       Random rnd = MatsimRandom.getLocalInstance();
216       Point p;
217       double x, y;
218       do {
219          x = g.getEnvelopeInternal().getMinX()
220          +  rnd.nextDouble()
221          * (g.getEnvelopeInternal().getMaxX() - g.getEnvelopeInternal().getMinX());
222
223          y = g.getEnvelopeInternal().getMinY()
224          + rnd.nextDouble() * (g.getEnvelopeInternal().getMaxY()
225          - g.getEnvelopeInternal().getMinY());
226
227          p = MGC.xy2Point(x, y);
228
229       } while (!g.contains(p));
230       Coord coord = new Coord(p.getX(), p.getY());
231       return coord;
232     }
233
234   /**************This is my method for geting point from coordinates***********/
235   private Coord createPoint(double X, double Y) {
236
237       Point p;
238       double x, y;
239
240          x = X;
241          y = Y;
242          p = MGC.xy2Point(x, y);
243
244       Coord coord = new Coord(p.getX(), p.getY());
245       return coord;
```

```
246     }
247
248
249
250
251   public Map<String,Geometry> readShapeFile(String filename, String attrString){
252
253
254     Map<String,Geometry> shapeMap = new HashMap<String, Geometry>();
255
256     for (SimpleFeature ft : ShapeFileReader.getAllFeatures(filename)) {
257
258         GeometryFactory geometryFactory= new GeometryFactory();
259         WKTReader wktReader = new WKTReader(geometryFactory);
260         Geometry geometry;
261
262         try {
263           geometry = wktReader.read((ft.getAttribute("type")).toString());
264           shapeMap.put(ft.getAttribute(attrString).toString(),geometry);
265
266         } catch (ParseException e) {
267           // TODO Auto-generated catch block
268           e.printStackTrace();
269         }
270
271       }
272     return shapeMap;
273   }
274
275   private Map<String,Coord> readFacilityLocations (String fileName, String home_or_work){
276
277     FacilityParser fp = new FacilityParser(home_or_work);
278     TabularFileParserConfig config = new TabularFileParserConfig();
279     config.setDelimiterRegex("\t");
280     config.setCommentRegex("#");
281     config.setFileName(fileName);
282     new TabularFileParser().parse(config, fp);
```

```
283     return fp.getFacilityMap();
284
285   }
286
287   private Coord findClosestCoordFromMap(Coord location, Map<String,Coord> coordMap){
288     Coord closest = null;
289     double closestDistance = Double.MAX VALUE;
290     for (Coord coord : coordMap.values()){
291       double distance = CoordUtils.calcEuclideanDistance(coord, location);
292       if (distance<closestDistance) {
293         closestDistance = distance;
294         closest = coord;
295       }
296     }
297     return closest;
298   }
299
300
301
302 }
303
304   class FacilityParser implements TabularFileHandler{
305     public String home_or_work;
306
307   FacilityParser(String workhome){
308     home_or_work = workhome;
309
310   }
311
312   private Map<String,Coord> facilityMap = new HashMap<String, Coord>();
313   CoordinateTransformation ct = new GeotoolsTransformation("EPSG:4326", "EPSG:32633");
314
315   @Override
316   public void startRow(String[] row) {
317     Double x;
318     Double y;
319     try{
```

```
320       if (home_or_work.equals("Home")){
321          x = Double.parseDouble(row[2]);
322          y = Double.parseDouble(row[1]);
323       }else{
324          x = Double.parseDouble(row[2]);
325          y = Double.parseDouble(row[1]);
326       }
327     Coord coords = new Coord(x,y);
328     this.facilityMap.put(row[0],ct.transform(coords));
329     }
330     catch (NumberFormatException e){
331       //skips line
332     }
333   }
334
335   public Map<String, Coord> getFacilityMap() {
336     return facilityMap;
337   }
338
339
340 }
```

Appendix B

Appendix 2

B.1 Conditional Probability Table

B.1.1 User 1

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.9	0.02	0.5	0.02	0.5	0.5	0.5

Table B.1: Node R

	R(0)	R(1)
A2(1)	0.01	0.2

Table B.2: Node A2

	R(0)	R(1)
A3(1)	0	0

Table B.3: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.6	0.02	0.5	0.02	0.5	0.5	0.5

Table B.4: Node A1

	A1(0)	A1(1)
A11(1)	0.01	0.3

Table B.5: Node A11

	A1(0)	A1(1)
A12(1)	0.01	0.03

Table B.6: Node A12

	A1(0)	A1(1)
A13(1)	0.01	0.34

Table B.7: Node A13

B.1.2 User 2

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.02	0.8	0.5	0.8	0.5	0.5	0.5

Table B.8: Node R

	R(0)	R(1)
A2(1)	0.01	0.9

Table B.9: Node A2

	R(0)	R(1)
A3(1)	0.01	0.9

Table B.10: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.02	0.6	0.5	0.2	0.5	0.5	0.5

Table B.11: Node A1

	A1(0)	A1(1)
A11(1)	0.08	0.9

Table B.12: Node A11

	A1(0)	A1(1)
A12(1)	0.02	0.01

Table B.13: Node A12

	A1(0)	A1(1)
A13(1)	0.08	0.55

Table B.14: Node A13

B.1.3 User 3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.9	0.9	0.5	0.02	0.5	0.5	0.5

Table B.15: Node R

	R(0)	R(1)
A2(1)	0.02	0.9

Table B.16: Node A2

	R(0)	R(1)
A3(1)	0.02	0.3

Table B.17: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.9	0.9	0.5	0.02	0.5	0.5	0.5

Table B.18: Node A1

	A1(0)	A1(1)
A11(1)	0.02	0.5

Table B.19: Node A11

	A1(0)	A1(1)
A12(1)	0.02	0.9

Table B.20: Node A12

	A1(0)	A1(1)
A13(1)	0.2	0.1

Table B.21: Node A13

B.1.4 User 3 model 1

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.9	0.9	0.5	0.02	0.5	0.5	0.5

Table B.22: Node R

	R(0)	R(1)
A2(1)	0.02	0.9

Table B.23: Node A2

	R(0)	R(1)
A3(1)	0.02	0.3

Table B.24: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.9	0.9	0.5	0.02	0.5	0.5	0.5

Table B.25: Node A1

	A1(0)	A1(1)
A11(1)	0.02	0.5

Table B.26: Node A11

	A1(0)	A1(1)
A12(1)	0.02	0.9

Table B.27: Node A12

	A1(0)	A1(1)
A13(1)	0.2	0.1

Table B.28: Node A13

B.1.5 User 3 model 2

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.9	0.02	0.5	0.02	0.5	0.5	0.5

Table B.29: Node R

	R(0)	R(1)
A2(1)	0.01	0.9

Table B.30: Node A2

	R(0)	R(1)
A3(1)	0.01	0.6

Table B.31: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.9	0.02	0.5	0.02	0.5	0.5	0.5

Table B.32: Node A1

	A1(0)	A1(1)
A11(1)	0.02	0.01

Table B.33: Node A11

	A1(0)	A1(1)
A12(1)	0.01	0.9

Table B.34: Node A12

	A1(0)	A1(1)
A13(1)	0.1	0.2

Table B.35: Node A13

B.1.6 User 3 model 3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
R1	0.5	0.9	0.02	0.5	0.9	0.5	0.5	0.5

Table B.36: Node R

	R(0)	R(1)
A2(1)	0.02	0.9

Table B.37: Node A2

	R(0)	R(1)
A3(1)	0.01	0.9

Table B.38: Node A3

t1	t1(0)				t1(1)			
t2	t2(0)		t2(1)		t2(0)		t2(1)	
t3	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)	t3(0)	t3(1)
A1	0.5	0.9	0.02	0.5	0.9	0.5	0.5	0.5

Table B.39: Node A1

	A1(0)	A1(1)
A11(1)	0.01	0.5

Table B.40: Node A11

	A1(0)	A1(1)
A12(1)	0.02	0.9

Table B.41: Node A12

	A1(0)	A1(1)
A13(1)	0.02	0.01

Table B.42: Node A13