An Agent Based Decision Support Framework for Healthcare Policy, Augmented with Stateful Genetic Programming

By
Marek Laskowski

A thesis submitted to
the Faculty of Graduate Studies
in partial fulfilment of
the requirements for the degree of
Doctor of Philosophy

Department of Electrical and Computer Engineering
Faculty of Engineering
University of Manitoba
Winnipeg, Manitoba

September 2010

Abstract

This research addresses the design and development of a decision support tool to provide healthcare policy makers with insights and feedback when evaluating proposed patient flow and infection mitigation and control strategies in the emergency department (ED). An agent-based modeling (ABM) approach was used to simulate EDs, designed to be tuneable to specific parameters related to specification of topography, agent characteristics and behaviours, and the application in question. In this way, it allows for the user to simulate various 'what-if' scenarios related to infection spread and patient flow, where such policy questions may otherwise be left "best intent open loop" in practice. Infection spread modeling and patient flow modeling have been addressed by mathematical and queueing models in the past; however, the application of an ABM approach at the level of an institution is novel. A conjecture of this thesis is that such a tool should be augmented with Machine Learning (ML) technology to assist in performing optimization or search in which patient flow and infection spread are signals or variables of interest. Therefore this work seeks to design and demonstrate a decision support tool with ML capability for optimizing ED processes. The primary contribution of this thesis is the development of a novel, flexible, and tuneable framework for spatial, human-scale ABM in the context of a decision support tool for healthcare policy relating to infection spread and patient flow within EDs . The secondary contribution is the demonstration of the utility of ML for automatic policy generation with respect to the

ABM tool. The application of ML to automatically generate healthcare policy in concert with an ABM is believed to be novel and of emerging practical importance. The tertiary contribution is the development and testing of a novel heuristic specific to the ML paradigm used: Genetic Programming (GP). This heuristic aids learning tasks performed in conjunction with ABMs for healthcare policy. The primary contribution is clearly demonstrated within this thesis. The others are of a more difficult nature; the groundwork has been laid for further work in these areas that are likely to remain open for the foreseeable future.

Acknowledgments

The author would like to acknowledge Dr. R.D. McLeod for his patience, support, and guidance during this lengthy doctoral process; B. Demianyk for his assistance in developing various aspects of the software project described herein; Dr. M. Friesen and B. Podaima for assistance in writing this document; Drs. Anderson, Alfa, Pedrycz, Peters, and Mukhi for influencing this work (in a positive way); D. Sanders, J. Kraut, and Jeff Allen for participating in various projects over the years (some related, some not); J. Aronsson and V. Okhmatovski for assistance parallelizing the project, and donating compute time, respectively. The author would also like to acknowledge CNPHI, Manitoba Hydro, Dr. L. Oppenheimer, and the Thorlakson foundation for financial support over the years. Apologies to anyone who has been left out, it has been a long and crazy ride.

Dedication

To Shannon, you are certainly my motivation.

Contents

Front Matter

List of Tables

List of Figures

List of Appendices

Chapter 1

Introduction

The time is approaching when Agent Based Models (ABM), Machine Learning (ML), and combined ABM-ML systems will be increasingly used to approach certain complex optimization problems formerly left to expert humans or "best intent open loop". In this work, Agent Based Model denotes a type of Multi-Agent System in which individual agents of varying types (entities capable of making decisions and interacting with their environment and other agents) along with their environment comprise the modeled system. The definition of Machine Learning for the purposes of this thesis is "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." [1]. The healthcare management literature describes a move *towards* "evidence-based decision making" calling for the incorporation of data from researchers as well as models when making healthcare policy decisions [2][3]. The discussion of barriers [3] to adoption of evidence based decision making, implies that the infrastructure for "closing the loop" is not yet in place. The alternative,

"Best intent open loop" is defined as an ad-hoc mode of incremental improvement to existing policy, an evolutionary optimization process without a formalized closed loop of performance feedback control. Ultimately, this thesis aims to prototype and demonstrate the utility of a decision support tool based on the aforementioned ABM-ML technology. Specifically, the application presented in this thesis is within healthcare policy decision support, a context defined by social interactions which are by their nature difficult to bound.

The primary contribution of this thesis is the development of a novel, flexible, and tuneable framework for spatially-oriented Agent Based Modeling. for the specific application is modeling infection spread and patient flow within a hospital emergency department (ED) in the context of a decision support tool for healthcare policy. The secondary contribution is the demonstration of the utility of Machine Learning for automatic policy generation with respect to the aforementioned healthcare policy decision support tool context. The application of Machine Learning to automatically generate healthcare policy in concert with an ABM is believed to be novel and of emerging practical importance. The tertiary contribution is the development and testing of a novel Machine Learning heuristic specific to the ML paradigm used: Genetic Programming (GP). This heuristic is meant to aid learning tasks performed in conjunction with ABMs for healthcare policy. The primary contribution is clearly demonstrated within this thesis. The secondary and tertiary contributions are of a more difficult nature; the groundwork has been laid in this thesis, providing clear direction for further work in these areas that are likely to remain somewhat open for the foreseeable future.

The emergency department system which forms the basis of the work consists of several agent types, and the system is characterized by rules governing the agent behaviours and interaction between these agents. Some agents enter the ED as patients presenting an Influenza Like Illness (ILI), while other patients are uninfected. Patients are eventually treated by nurses and doctors according to their condition. Patients are discharged (admitted to a hospital ward, or sent home) once their treatment is complete. Discharged patients may have acquired the ILI causing virus while visiting the ED, others may have been infected prior to visiting the ED, and others may be uninfected. Signals which characterize this system are the flow of patients entering the system as well as the spread of influenza virus, albeit the latter would have to be mined or estimated from data in the future once patients manifest symptoms.

Specific challenges addressed by this thesis include a high degree of complexity [4] owing to the inherently dynamic social nature of the system as well as incomplete or insufficient data regarding ED state. A further challenge is designing a modeling framework which is built from the ground up to make use of increasingly detailed real data that is being collected but currently not applied in any systemic way to the development of the type of simulation framework proposed.

The problem undertaken by this thesis is to design a decision support tool to provide clinicians, hospital administrators, and other health policy makers with insights and feedback when evaluating proposed patient flow and infection mitigation and control strategies in the ED which might otherwise be left “best intent open loop”. A conjecture of this thesis is that such a tool should be augmented with Machine Learning technology to assist in performing optimization or search tasks in which patient flow and infection

spread are signals or variables of interest. Therefore this work seeks to design a decision support tool with machine learning capability for optimizing emergency department processes. Section 2.5.2 will list many machine learning successes in limited or toy problem domains, and it will be brought to bear on more difficult problems inclusive of social dynamics and concomitant unpredictable human behaviour in Chapter 7.

Canada's publicly funded healthcare system has long been a well known source of national pride. In 2009 Canada was estimated to spend 183.1 billion dollars or 11.9% of its GDP [5] on healthcare. Without overstating the importance of this investment, potentially any small improvement in the effectiveness of the healthcare system could save billions of dollars. In 2009, 27.8 percent of healthcare expenditures are estimated to be spent on hospitals [6], with the ED being at the front line of healthcare during any sort of public health emergency, for example a pandemic. Furthermore, patient flow bottlenecks are well known and long standing healthcare access issues[7][8][9]. Issues of infection control are being brought to the forefront during an increased time of pandemic awareness by policy makers in the public sector [10].

1.1 Overview of Related Work

Healthcare delivery systems have been described as complex adaptive systems that are appreciatively difficult to analyze for the purpose of improving overall operational performance and efficiencies [11]. One can consider them to exhibit properties of a highly connected network of formal and informal nodes that are by nature adaptive due to "learning" mechanisms (decisions) derived by the collaborations of observers (e.g.,

healthcare providers, managers, policy makers, stakeholders, and government officials). Due to the interconnectedness, and therefore interdependence of these nodes, the activity of one node has the potential to affect the behaviour of the entire healthcare network. This makes it intrinsically difficult for operations managers to assess and predict the affects of intervention and policy changes as their implementations have system-wide consequences. As such, healthcare delivery systems are inherently open and reveal behaviours of non-linear dynamics and stochastic processes – requiring the employment of a myriad of statistical and mathematical techniques (including both simulation and modeling) for the purpose of analysis.

Providing both timely and effective healthcare services in an acute care or ambulatory setting is of critical importance and have far-reaching implications in terms of societal, organizational (reputation), and operational wellness. Hence, there is much emphasis and pressure placed on organizations when it comes to improving front-line healthcare services – in particular, the level of patient safety and quality of care in healthcare delivery units such as emergency departments, urgent care, and ambulatory care centres. In an attempt to improve general decision making in healthcare delivery, decision support algorithms and software systems are being considered for broad implementation, as practical support tools for clinicians, operations managers, policy-makers, and vested stake-holders.

1.1.1 Simulation and Decision Support tools for Healthcare Systems

There are plethora of modeling and simulating healthcare systems reported in the literature [12]. Many of these have been investigated for their effectiveness as decision support tools (DST) in healthcare delivery, and are based primary on operations management (queuing theory) [13][14][15][16], Discrete Event Simulation (DES) [17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34], Markov processes [35], expert systems [36], mathematical (equation based) models often used to study specific phenomenal such as infection spread [37], and Systems Dynamics (SD) modeling. Both DES and SD have adoption challenges and have difficulty lending themselves to healthcare simulation: DES tends to require patient flow derived detail-level models; and, SD rely on aggregate models, not derived from patient flows but patient classes [11]. They also require limiting assumptions in order to make the modeling realistic, but nevertheless lack the means of incorporating 'adaptation' and mimicking 'human behaviour' [38][39]. If these assumptions do not hold, then the models generated may simply not hold true. As the development of a framework for simulating healthcare delivery necessitates the modeling of complex adaptive systems, then tools have to be designed (or modified) to reflect this [4][40]. Given the dynamic nature of healthcare delivery, and its pervasive interacting complex processes, Agent-based Modeling (ABM) has been considered as a viable candidate (and has shown remarkable promise) for the realistic capturing of these behaviours, and thus in the architecting of suitable decision support tools.

1.1.2 Agent-based Decision Support for Healthcare Engineering

Considerable work has been done with ABMs in the area of logistics, economics, business, and tactical decision making and support systems. Simulations employing ABMs allow one to identify losses and test mitigation and operational procedures, and model risk in general [11]. Although there have been a large number of ABMs developed for applications in social behaviour modeling (and social science disciplines) they have also been applied to epidemic modeling in human populations, as this is an important unsolved problem which garners a significant amount of public health and public policy attention [41][42]. To date, however, relatively little work has been done on applying ABMs specifically to healthcare operations management and healthcare delivery [11][39] [43][44][45][46][47][48][49][50][51][52][53]. One particular paper [11], outlines one of the most comprehensive works in the emerging area of applying ABMs to healthcare operations management and healthcare delivery.

Agent based modeling (ABM) and simulation [54] has the capacity to offer both a flexible and scalable framework that can embody an entire healthcare system – including its internal network complexities. As ABM enables exploration of a system behaviour over a broad range of parameters it may provide crucial pre-implementation insight into the impact of a new system or technology on workflow and other variables important to providing better healthcare delivery [45]. It can do so with effective representational plausibility, augmenting more traditional approaches based almost entirely on inductive and deductive reasoning. As a result, effective decision support tools can be developed for healthcare operations management using ABMs having purposeful utility, with the

means to incorporate and test functional improvements on an incremental basis. Furthermore, ABMs can also evolve along with the experiences of observers (experts) and reveal emergent behaviours [55] not otherwise possible with more conventional discrete event simulators (DES).

In an ABM, a system is modeled from the ground up: describing a system from the perspective of its constituent parts (microscopic view), in order to build an aggregate (macroscopic) picture of the whole. Systems are modeled as a collection of agents, each of which are autonomous decision-making entities able to assess their situation, make decisions, and compete with one another on the basis of a set of rules. In that way, an ABM can be described as system modeling based on individual agents and their behaviours and interactions, yet an ABM's conceptual depth is derived from its ability to model emergent behaviour that may be counterintuitive or, at minimum, discern a complex behavioural whole that is greater than the sum of its parts [38]. Hence, one can consider ABM as a tool employing a multiplicity of interacting agents to 'reveal' emergent phenomena that is collectively more that the constituent properties of the individual agents themselves.

The benefits of ABM lie in the technique's ability to capture emergent phenomena, which also leads to further benefits: An ABM provides a natural description of a system that can be calibrated and validated by representative expert agents, and an ABM is flexible enough to tune the complexity of agent behaviour, rationality, learning, and rules of interactions [56]. There are a number of characteristics of agents and systems for which ABM is a particularly powerful modeling technique, and which simultaneously make more mathematically-based approaches difficult or intractable [56][57]. These

system features include individual agent behaviour that is complex, non-linear, potentially stochastic, and may exhibit memory or path-dependence, agent interactions that are heterogeneous, instances in which averages are not meaningful, and systems that are best described by activities and interactions rather than by processes. All of these features are necessary when one attempts to represent and mimic human-to-human and human-to-inanimate object interaction, and model within a complex dynamic environment. This functionality is especially important for corroboration when conducting experimentation and trials in Human Factors Engineering (HFE), and applying such disciplines as Failure Mode and Effects Analysis (FMEA), and Root Cause Analysis (RCA). A decision support tool incorporating these features derived by ABM has far reaching implications for improving healthcare simulation.

There are many instances reported in the literature today whereby ABM has demonstrated its viability and efficacy, and holds much promise in further improving modeling methodologies beyond that previously thought attainable with more conventional means. However, there are caveats when employing ABMs: in order to 'win over' the more conventional methodologists and "for ABM to meet its promise, practitioners should resist the temptation to overstate the implications of model outcomes, carefully design agent-based models with an eye toward ecological validity, and provide precise specifications and results based on families of models subjected to repeated tests" [58].

1.1.3 Agent-based Decision Support Tools for Emergency Department Management

One of the most desirable features of a decision support tool for emergency department (ED) management utilizing ABM is that is will easily accommodate real-world scenarios and variations therein. For instance, variations in patient arrival rate, based on time of day, day of the week, time of year, or variation between individual EDs can easily be implemented in an ABM simulator. For many discrete event simulators these sorts of variations are intractable have been assumed to be of a uniform rate. When such information is available, it is possible to estimate arrival rates of patients for each triage level in a simulated ED and further demonstrate the viability of the ABM approach. While predictions based on modeling and simulation is extremely difficult and potentially error prone, its confidence can be enhanced as predictions are tracked against real data. As with the other ABM applications, a model of EDs augmented with available empirical data offers advantages over "best intent open loop" policy decision making.

In order for a healthcare delivery decision support tool to find acceptance in an ED environment, and confidently be deployed in practice (e.g., to reducing patient waiting times in EDs), it would have to utilize more accurate, reliable, purposeful, representational, and operational ABMs. Thus, from a research and development perspective, there is an emphasis to:

- Refine and extend existing agent-based modeling frameworks (tools), with specific applications to modeling and optimization of patient access and patient waiting times in emergency departments;

- Use the refined modeling tools to comparatively model existing and proposed patient access and patient care strategies within the emergency department; and,
- Extend the modeling tools to other healthcare areas (beyond the emergency department) and/or to modeling patient access and flow between multiple health care facilities.

Based on the flexible, scalable, and modular nature of ABMs, many further refinements to an ED management decision support tool can be specified, formalized, tested, and validated. Specific development areas for exploration include:

- Enhancing the sensitivity and accuracy of the models and allowing for increased variability in input parameters. Input parameters include the topology of the emergency department, as well as agents' characteristics, behaviours, and interactions with one another;
- Augmenting models with real data on agent characterization, behaviours, and interactions (vs. heavy reliance on simulated data); and,
- Modeling comparative patient access and patient care strategies.

One of the first objectives in refining the ABM as a viable decision support tool for healthcare operations management, and establishing validity of the approach, will be to close the 'gap' between simulated data to actual data. Within the context of policy evaluation, the modeling of strategic initiatives demonstrated in other contexts should be investigated. For example, a number of initiatives have been introduced elsewhere that may have variations that could be implemented in a decision support system (DSS) within the local context [59].

1.1.4 Validation of Agent-based Decision Support Tools for Emergency Department Management

Validation of agent-based decision support tools are derived through multiple methods, including expert review (clinicians), comparison to real baseline data (experientially-derived), as well as statistical analysis. Model data from a significant number of simulation instances and scenarios can be analyzed subject to multivariate statistics, to provide some level in confidence in the significance of and relationships between the variables under investigation. The systems that are of interest for modeling tend to be highly nonlinear and difficult to analyze, although there are a number of statistical tools available that can be utilized. As such, the ABM for decision support is merely a tool that is descriptive and still requires vetting from an experiential knowledge base.

There are many concomitant theoretical and practical outcomes of ABM simulation for decision support in an emergency department, among them are a better understanding of the comparative impacts of patient care and patient access policies and practices. These include matters related to clinical operations management [60], clinical information flow [61], and treatment modalities [62]. Others relate to resource allocation and supply chain management for the distribution and utilization of medical products, such as medical equipment, devices, supplies, and medications. From a human resource management perspective, investigation into efficient and fair policy and practice for medical personnel and staff can also be assessed, evaluated, challenged, tested, and circumvented with a healthcare delivery human factors decision support tool based on ABMs. An ancillary objective of employing a decision support tool is to extend and

refine the existing ABM framework for a specific and highly detailed application, with the outcome of improved understanding of system dynamics, risk, and mitigative strategies or operational policies within a healthcare setting. Various "what-if" scenarios can be simulated, and thereby evaluated, with the outcome to offer empirical support to practitioners, planners, and policy-makers in the respective instance context.

Holding informal focus groups as well as one-on-one interviews with practitioners from emergency departments serves as a demonstration of the requirements-gathering process in developing patient care scenarios and patient flows. This type of data collection is critical in improving individual department models in terms of their physical components, the resources allocated (doctors, nurses, beds, rooms, administrators, equipment), as well as in better understanding use-case scenarios. Consultation with expert agents (healthcare practitioners) is also a validation technique within the ABM approach. It is this level of detail that the development and refinement of a more complete ABM will address, in order for the full utility of the ABM to be realized. Requirement gathering of this nature is critical for validation of the ABM and comprises a necessary component for the development of a useful decision support tool.

There are many examples demonstrating the utility of ABM techniques relative to competing and complementary equation-based techniques (implemented in discrete event simulation, DES). However, as with all emerging technologies there are a number of extensions and research challenges that have to be recognized and overcome, which will serve to enhance the overall power and utility of an ABM. One of the most persuasive qualities of the ABM is its ability to close the communication gap between the healthcare researchers and code developers. A by-product of ABM development being closely

coupled to a practitioner's view is that verification is more easily undertaken, as a consequence of the knowledge translation during the decision support implementation and software tool development phases. Figure 1.1 illustrates the general relationship between the modeling and simulation methodologies mentioned.

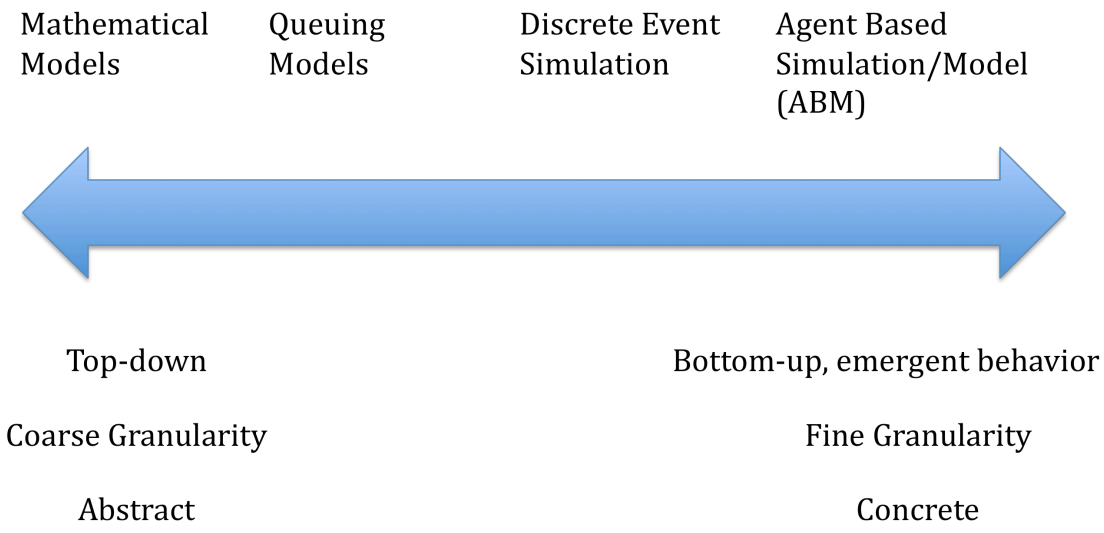

Figure 1.1. Comparison of various modeling and simulation methodologies

1.1.5 Decision Support Tools for Emergency Department Management with Machine Learning

Relatively little work has been done applying Machine Learning within Decision Support Tools for Emergency Department Management. Two notable examples of basic Machine Learning applications within such tools include optimization of staffing schedules using an Discrete Event Simulation and a Genetic Algorithm [27], and training of a neural-network based metamodel for rapid interpolation of Discrete Event Simulation results [63]. The machine learning task involves the induction of a patient flow and treatment

policy to increase patient flow and minimize airborne disease transmission. Genetic Programming [1] is eventually chosen as the specific machine learning approach for this task.

1.2 Organization of the work

The remainder of this chapter outlines how this thesis will address the shortcomings apparent in the related work – that is, the lack of decision support tools for healthcare suited to both data driven what-if scenarios, positioned to leverage increasingly detailed data, and amenable to Machine Learning augmentation for automated generation of healthcare policy. The concept map in Figure 1.2 illustrates the relationships between the concepts presented in the remaining chapters.

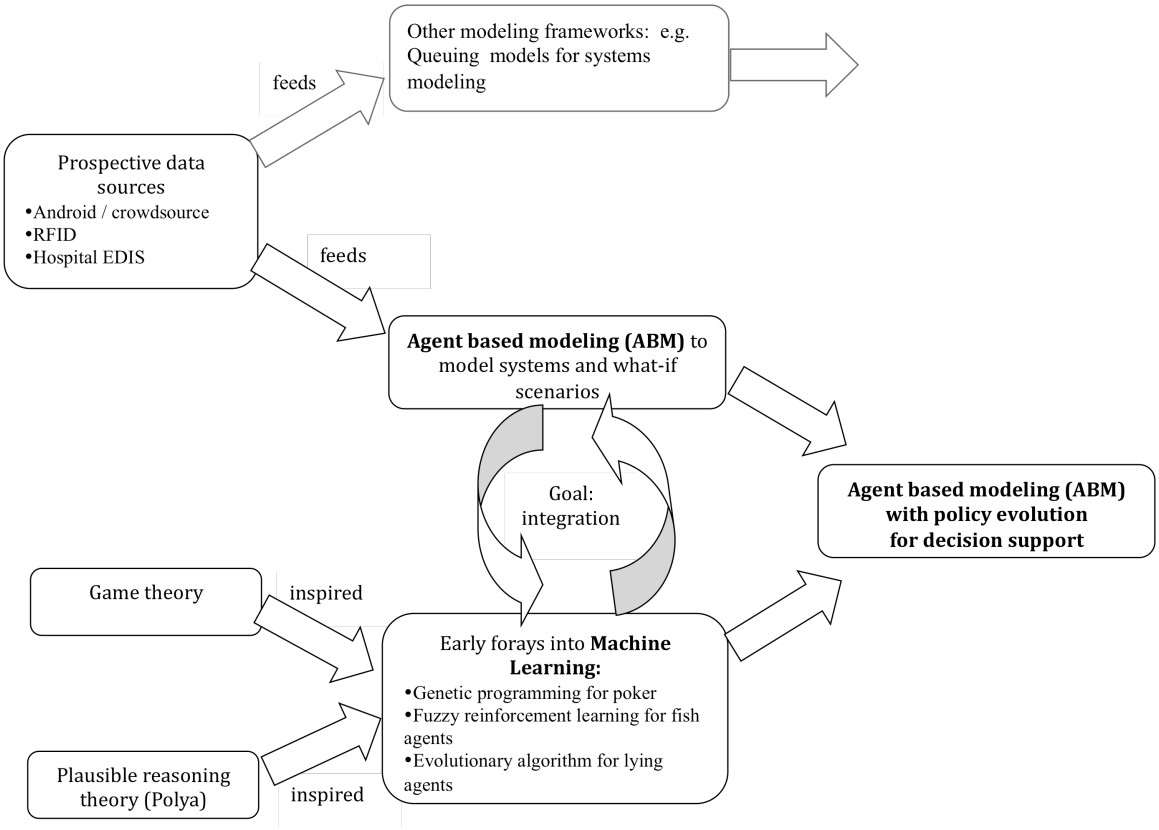

Figure 1.2. Concept Map for this Thesis Document.

This author's early work was inspired by Game Theory and Polya's Plausible Reasoning Theory. It was a small but surprising discovery that a simple evolutionary algorithm could evolve an agent lying policy that unexpectedly invaded a population of policies introduced by researchers [64]. The ultimate implication of this was that machine evolved polices within Agent Based Models could be used to support or supplement human decision making in complex domains by generating novel policies. This insight was carried forward in the work. In other words, the early work illuminated a solution approach that demonstrated a natural fit with the increasing interest in healthcare

policy decision support within the realm of pandemic preparedness beginning in approximately 2007.

Chapter 2 is a survey of Multi-Agent Systems (MAS) and specifically Agent Based Modelling and Simulation frameworks, beginning with a taxonomy of MAS. Strengths and weaknesses of Agent Based Models are discussed as well. A comparison the features of many prominent ABM frameworks leads to a list of desirable feature set for emerging ABM frameworks such as the one discussed throughout this thesis. Chapter 2 concludes by surveying Machine Learning applications within MAS, also referred to as Distributed Artificial Intelligence (DAI), and choosing a candidate approach for application in Chapter 7.

Chapter 3 discusses the early work focused on modeling patient flow within a hospital ED using an Agent Based Model. It is now possible to model complex healthcare phenomena in terms of their constituent parts, that is, agents and rules governing behaviours of and interactions between agents. In contrast to purely mathematical models which abstract away complexity and impose an artificial structure on the problem, models structured as a collection of rules governing interactions of agents are able to capture more social details and a higher degree of realism than ever before. Instead of only characterizing system output, it becomes possible to ask new types of questions concerning complex healthcare systems, such as "what-if" questions concerning infection spread within hospital emergency departments. Development of such models eventually enables automated policy generation as a means of decision support in complex healthcare contexts which may otherwise remain "best intent open loop".

Chapter 4 expands on data collection technologies implicitly required in Chapter 3. Integration of such data sources will be necessary to validate and ultimately enhance the fidelity of these ABMs. The visualization component of the ABM, in addition to its close resemblance to the modeled system, are suggested as a potential role for the ABM as a means of facilitating communication between modellers and healthcare practitioners.

Chapter 5 contrasts ABMs with complementary modeling techniques such as Queuing Models. The utility of integrating enhanced data sources, such as those introduced in Chapter 4, is discussed for both ABM and complementary models. Furthermore, Chapter 5 also lays the groundwork for, and suggests the utility of two Machine Learning tasks with respect to an ABM-ML based decision support tool, the chapter's main contribution which will provide direction for the later chapters. Chapter 6 further extends the ABM to model infection spread within the ED environment. The simulator at this point has been extended to consider numerous "what-if" scenarios or investigations, the results of which are presented and discussed. This ability to exercise "what-if" scenarios will be used to evaluate automatically generated policies vis-a-vi infection spread is key to achieving automated policy generation for healthcare, a stated goal of this thesis.

In Chapter 7, the promise of a combined ABM-ML decision support tool capable of automatically generating policy within the modeled ED is fulfilled. More precisely, an evolutionary process is the specific means of achieving automated policy generation. The objective of this policy is to optimize patient flow and to mitigate the spread of an Influenza Like Illness causing virus throughout the ED. This application moves the chosen ML paradigm, Genetic Programming, out of the toy-problem domain where

verification of ML performance is considerably simpler. The developed prototype decision support tool is used to demonstrate the utility of such a system. The chosen ML paradigm, Genetic Programming is said to be "mooreware" [65], that is able to take advantage of the exponentially increasing computing power implied by Moore's Law. Both these factors strongly add credibility to the widespread use of such ABM-ML systems within decision support tools in the near future. This chapter also presents a novel GP heuristic which adds the concept of "states" to GP, as suggested by the various states underlying the ABM, implying a Markov Decision Process (MDP).

In order to demonstrate that automated generation of policy in a non-toy domain was possible, several significant and concurrent steps were undertaken. For example, although data sources have been identified to feed the models developed in this thesis, the full power and utility of the techniques presented herein will be realized as the technologies become available to deliver accurate and detailed information about the ED, and political barriers related to information access are overcome. As mentioned, related work lacks the data centrism and amenability to Machine Learning needed to accomplish the overall goal of the thesis. As such, much of this work was developed with very little reliance on previous efforts.

As with any early foray into a unique area, this work may incite scepticism in some readers. We stand at the threshold of machine augmented decision support for complex real-world problems. Throughout this thesis, the primary concern is developing the framework for machine augmented decision support systems, on which a body of applications can be built, to the point that such systems are used routinely in complex decision making scenarios which now remain "best intent open loop" or addressed

through approaches that lack the flexibility and realism of ABMs. However, it may take years before this is commonplace.

Owing to the nature of this thesis where several chapters have been published or submitted for publication, some of the material presented in earlier chapters, particularly with respect to descriptions of the ABM, is re-iterated in later chapters. Chapters 3, 4, 5, and 6 represent a progression of ABM capabilities at different stages of development. Therefore, the original papers have been edited to reduce the amount of repetitive material while emphasizing changes in ABM features and capabilities throughout the development process.

1.3 References

[1] T. Mitchell, *Machine Learning*. Boston, MA: McGraw Hill, 1997.

[2] "CHRSF Knowledge Transfer: Decision Support: A New Approach to Making the Best Healthcare Management and Policy Choices," *HealthCare Quarterly*, (Online). http://www.longwoods.com/content/18918 (Accessed: September 2010).

[3] "Creating a Culture of Evidence-Based Decision Making," *Health Canada*, (Online). http://www.hc-sc.gc.ca/hcs-sss/pubs/renewal-renouv/1997-nfoh-fnss-v2/legacy_heritage5-eng.php (Accessed: September, 2010).

[4] M. Smith, C. Feied. The emergency department as a complex system. [Online]. Available at http://necsi.org/projects/yaneer/emergencydeptcx.pdf. [Accessed: May 10, 2010].

[5] Canada. Canadian Institute for Health Information. *Total Health Expenditure, Canada, 1975 to 2009-Summary.* [Online]. Available: Quick Stats, http://secure.cihi.ca/cihiweb/en/media_20091119_tab1_e.html. [Accessed: May 10, 2010].

[6] Canada. Canadian Institute for Health Information. *Total Health Expenditure by Use of Funds, Canada, 1975 to 2009-Current Dollars.* [Online]. Available: Quick Stats, http://secure.cihi.ca/cihiweb/en/media_20091119_tab2_e.html. [Accessed: May 10, 2010].

[7] L. Keith, "Liberal platform promises to reduce emergency-room wait times," *Winnipeg Free Press*, August 27, 2007.

[8] G. Giroday, "Hospital probes how man died after 34-hour wait in ER," *Winnipeg Free Press*, September 23, 2008. [Online]. Available: http://www.winnipegfreepress.com/local/story/4229550p-4870606c.html. [Accessed September 24, 2008].

[9] J. Skerritt, "Woman, 82, died in St. Boniface emergency," *Winnipeg Free Press*, September, 26, 2008. [Online]. Available: http://www.winnipegfreepress.com/breakingnews/story/4231139p-4872336c.html. [Accessed March 10, 2009].

[10] Canada. Canadian Network for Public Health Intelligence. *The Canadian Pandemic Influenza Plan for the Health Sector.* [Online]. Available: http://www.phac-aspc.gc.ca/cpip-pclcpi/index-eng.php [Accessed: May 10, 2010].

[11] Kanagarajah, A.K.; Lindsay, P.A.; Miller, A.M.; and Parker, D.W., "An exploration into the uses of agent-based modeling to improve quality of health care," *Int. Conf. on Complex Systems*, Boston, MA, 2006.

[12] Jun, J.B.; Jacobson, S.H.; and Swisher, J. R., "Application of discrete-event simulation in health care clinics: a survey," *The Journal of the Operational Research Society*, vol. 50, no. 2, pp. 109-123, Feb. 1999.

[13] Vos, L.; Groothuis, S.; and van Merode, G.G., "Evaluating hospital design from an operations management perspective," *Health Care Management Science*, vol. 10, no. 4, pp. 357-364, Dec. 2007.

[14] Green, L.V., "Using operations research to reduce delays for healthcare," In *Tutorials in Operations Research*, eds. Zhi-Long Chen and S. Raghavan, Hanover, MD: INFORMS, pp. 1-16, 2008.

[15] Georgievskiy, I.; Georgievskaya, Z.; and Pinney, W., "Using Queuing analysis and computer simulation modeling to reduce waiting time in the hospital admitting department," Aug. 2009. [On-line] http://www.flexsim.com/products/healthcare/docs/Reduce_Admissions_wait_times.pdf

[16] Shiver, J.M.; and Eitel, D., Optimizing Emergency Department Throughput: Operations Management Solutions for Health Care Decision Makers, Pub.: Taylor & Francis Books Ltd., Dec. 2009.

[17] Badri, M.; and Hollingsworth, J., "A simulation model for scheduling in the emergency room," I*nt. Journal of Operations & Production Management*, vol. 13, no. 3, pp. 13 – 24, 1993.

[18] Benneyan, J.C., "An introduction to using computer simulation in healthcare: patient wait case study," *J Soc Health Syst.*, vol. 5, no. 3, pp, 1-15, 1997.

[19] Price, R.N.; and Harrell, C.R.; "Healthcare simulation modeling and optimization using MedModel," In *Proc. of the 31st Winter Conference on simulation*, pp. 215-219, 1999.

[20] Connelly, L.G.; *Acad Emerg Med.*, "Discrete event simulation of emergency department activity: a platform for system-level operations research," vol. 11, no. 11, pp. 1177-1185, Nov. 2004.

[21] Sinreich, D.; and Marmor, Y., "A simple and intuitive simulation tool for analyzing emergency department operations" *Proc. of the 36th Winter conference on simulation*, pp. 1994 - 2002, 2004.

[22] Sinreich, D.; and Marmor, Y., "Emergency department operations: the basis for developing a simulation tool," *IIE Transactions*, vol. 37, no. 3, pp. 233 – 245, Feb. 2005.

[23] Jacobson, S.H.; Hall, S.H.; and Swisher, J.R., Discrete-Event Simulation of Health Care Systems, Int. Series in Operations Research & Management Science, Patient Flow: Reducing Delay in Healthcare Delivery, Chapter 8, Springer, vol. 9, Oct.11, 2006.

[24] Eldabi, T; and Young, T.; "Towards a framework for healthcare simulation," *Proc. of the 39th Winter conference on simulation: 40 years! The best is yet to come*," pp. 1454-1460, Dec. 2007.

[25] Kuljis, J.; Paul, R.J.; and Stergioulas, L.K.; "Can health care benefit from modeling and simulation methods in the same way as business and manufacturing has?," *Proc. of*

the 39th Winter conference on simulation: 40 years! The best is yet to come, pp. 1449-1453, 2007.

[26] Su, S.; and Shih, C-L.; "Modeling an emergency medical services system using computer simulation," *Int J Med Inform.*, vol. 72, pp. 57-72, Dec. 2007.

[27] Yeh, J-Y.; Lin, W-S.; "Using simulation technique and genetic algorithm to improve the quality care of a hospital emergency department," *Expert Systems with Applications: An International Journal*, Vol. 32, no. 4, pp. 1073-1083, May 2007.

[28] Georgievskiy, I.; Georgievskaya, Z.; and Pinney, W.; *Proc. of the Business & Health Administration Conference (BHAA)*, Chicago, Illinois, April 2-4, 2008.

[29] Medeiros, D.J.; Swenson, E.; and DeFlitch, C.; "Improving patient flow in a hospital emergency department," In *Proc. of the 40th Winter Conference on Simulation*, pp. 1526-1531, 2008.

[30] Efe, K.; Raghavan, V.; and Choubey, S., "Simulation modeling movable hospital assets managed with RFID sensors," *Proc. of the 2009 Winter Simulation Conference*, pp. 2054-2064, Dec. 2009.

[31] Marmor, Y.N.; Wasserkrug, S.; Shtub, A., "Toward simulation-based real-time decision-support systems for emergency departments," *Proc. of Winter Simulation Conference (WSC'09)*, Dec. 13-16, pp. 2042-2053, 2009.

[32] Wasserkrug, S.; Greenshpan, O.; Marmor, Y.N.; Carmeli; B.; Vortman, P.; *et al.*, "InEDvance: advanced IT in support of emergency department management," In *Proc. of the 7th Conference on Next Generation Information Technologies and Systems, NGITS*, Haifa, Israel, June 16–18, 2009.

[33] Zeltyn, S.; Carmeli, B.; Geenshpan, O,; Mesika, Y.; Wasserkrug, S, *et al.*, "Simulation-based models of emergency departments: Operational, tactical and strategic staffing, *Sub. to ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 2009.

[34] L. Patvivatsiri, "A Simulation Model for Bioterrorism Preparedness in an Emergency Room", *Proceedings of the 2006 Winter Simulation Conference*, pp.501-508, 2006.

[35] Karnon, J.; "Alternative decision modelling techniques for the evaluation of health care technologies: Markov processes versus discrete event simulation," *Health Econ.*, vol. 12, no. 10, pp. 837-848, Oct. 2003.

[36] Graber, M.A.; and Vanscoy, D., "How well does decision support software perform in the emergency department?," *Emergency medicine journal*, vol. 20, no5, pp. 426-428, 2003.

[37] C. van den Dool, M.J.M. Bonten, E. Hak, J.C.M. Heijne, and J. Wallinga, "The effects of influenza vaccination of health care workers in nursing homes: Insights from a mathematical model," PLoS Medicine, vol. 5, no. 10, e200 doi:10.1371/journal.pmed.0050200.

[38] Bonabeau, E., "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences*, 99 (3): pp. 7280-7287, May 2002.

[39] Stainsby; H.; Taboada, M.; and Luque, E., "Towards an Agent-Based Simulation of Hospital Emergency Departments," *International Conference on Services Computing, 2009. SCC '09. IEEE* , pp.536-539, 21-25 Sept. 2009.

[40] Eldabi, T.; Paul, R.J.; Young, T., "Simulation modelling in healthcare: reviewing legacies and investigating futures," *Journal of the Operational Research Society*, vol. 58, no. 2, pp. 262-270, Feb. 2007.

[41] Epstein, J.M., "Artificial society: Getting clues on how a pandemic might happen by creating a huge model of the United States," The Brookings Institution, April 2 2008. [Online]. www.brookings.edu/interviews/2008/0402_agent_based_epstein.aspx.

[42] National Institute of General Medical Sciences, "MIDAS: models for infectious disease agent study," 2010. [Online]. Available: https://www.epimodels.org/midas/about.do

[43] Nealon, J.L.; and Moreno, A., "Agent-based applications in health care," *Applications of Software Agent Technology in the Health Care Domain* (John L. Nealon and Antonio Moreno, Eds.), Whitestein Series in Software Agent Technologies, Birkh¨auser Verlag, Basel, Switzerland, pp. 3-18, 2003.

[44] Christiansen, J. H.; and Campbell, A. P, "HealthSim: An Agent-Based Model for Simulating Health Care Delivery," Argonne National Laboratory, Chicago, Illinois, 2003.

[45] Poynton, M.R.; Shah, V.M.; BeLue, R.; Mazzotta, B.; *et al.* "Computer terminal placement and workflow in an emergency department: An agent-based model," 2007.

[46] Blachowicz, D.; Christiansen, J.H.; Ranginani, A.; and Simunich, K.L., "How to determine future HER ROI: Agent-based modeling and simulation offers a new alternative to traditional techniques," *J Healthcare Information Management*, vol. 22. no. 1, pp. 39–45, 2008.

[47] Daknou, A.; Zgaya, H.; Hammadi, S.; and Hubert, H., "Toward a multi-agent model for the care of patients at the emergency department," *Pro. of the 10th WSEAS Int. Conf.*

on Mathematical Methods, Computational Techniques and Intelligent Systems (MAMECTIS '08), Corfu, Greece, pp. 264-269, Oct. 2008.

[48] Daknou, A.; Zgaya, H.; Hammadi, S.; and Hubert, H., "Agent based optimization and management of healthcare processes at the emergency department," *Int. Journal of Mathematics and Computers in Simulation*, vol. 2. (3), pp. 285-294, Nov. 2008.

[49] Hutzschenreuter, A.K.; Bosman, P.A.N.; Blonk-Altena, I.; van Aarle, J.; La Poutr´e, J.A., "Agent-based patient admission scheduling in hospitals," *In: Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008) – Industry and Applications Track*, pp. 45-52, 2008.

[50] Jones, S.S., and Evans, R.S., "An agent based simulation tool for scheduling emergency department physicians," *Proc. of AMIA Ann. Symp*, pp. 338-342, 2008.

[51] Paranjape, R; and Sadanand, A, (Eds.), Multi-agent systems for healthcare simulation and modeling: applications for system improvement, Pub: Medical Information Science Reference, USA, Aug. 2009.

[52] Wang, L., "An agent-based simulation for workflow in Emergency Department," *Systems and Information Engineering Design Symposium*, *2009. SIEDS '09*, pp.19-23, April 24-24, 2009.

[53] Isern, D.; Sáncheza, D.; and Morenoa, A., "Agents applied in health care: A review," *International Journal of Medical Informatics*, vol.79, Issue 3, pp. 145-166, March 2010.

[54] Macal, C.M; and North, M.J, "Agent-based modeling and simulation," *Proc. of the 2009 Winter Simulation Conference*, (M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, Eds.), pp. 86-98, 2009.

[55] Gordon, R., "The emergence of emergence: a critique of 'design, observation, surprise!'," *Rivista di Biologia*, 93 (2), pp. 349-356, 2000.

[56] Axelrod, R., The Complexity of Cooperation: Agent-based Models of Competition and Collaboration, Princeton, NJ: Princeton University Press, 1997.

[57] Epstein, J.M.; and Axtell, R.L., Growing Artificial Societies: Social Science from the Bottom Up, Cambridge, MA: MIT Press, 1996.

[58] Jones, G.T, "Agent-based modeling: use with necessary caution," *American Journal of Public Health*, vol. 97, no. 5, pp. 781-782, May 2007.

[59] Cooke, M., "Ambulances and overcrowded emergency departments," Dec. 14, 2006. [Online].www2.warwick.ac.uk/fac/med/research/hsri/emergencycare/research/ambulance/amb_ed_nopics.ppt.

[60] Foster , D.; Mcgregor, C.; and El-masri, S., "A survey of agent-based intelligent decision support systems to support clinical management and research," In: G. Armano, E. Merelli, J. Denzinger, A. Martin, S. Miles, H. Tianfield and R. Unland, Editors," 1st Intl. Workshop on Multi-Agent Systems for Medicine, Computational Biology, and Bioinformatics (AAMAS), 2005.

[61] Franck, M.; Göbel, M.; and Friesdorf, W., "Agent-based Information Management in an Emergency Room," In H. Strasser, K. Kluth, H. Rausch & H. Bubb (eds.), *Quality of Work and Products in Enterprises of the Future. Stuttgart: Ergonomia,* pp. 685-688, 2003.

[62] J. Huang, N.R. Jennings, and J. Fox, "An agent-based approach to health care management," *Internat. J. Appl. Artificial Intelligence*, vol. 9 (4), pp. 401–420, 1995.

[63] R.A. Kilmer, A.E. Smith and L.J. Shuman, "An emergency department simulation and a neural network metamodel,", *Journal of the Society for Health System*, vol. 5, no. 3, 1997, 63-79.

[64] M. Laskowski and S. McGrath, “Effects of lying in reputation-based multi-agent systems,” in Elect. and Comput.Eng. 2005: Canadian Conf. on, 2005, pp. 1014-1018.

[65] J.R. Koza, et al., *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. New York, NY: Springer, 2005.

Chapter 2

Introduction to Distributed Artificial Intelligence and Agent Based Models: with Emphasis on Learning in Multi-Agent Systems

This chapter overviews Multi-Agent Systems, specifically Agent Based Models, and presents a taxonomy for comparing Multi-Agent Systems (section 2.1). The strengths and weaknesses of the Multi-Agent Modeling paradigm are then discussed (section 2.2). Ultimately, comparisons are made between existing and emerging platforms for Multi-Agent Simulation (section 2.3) leading to an outline for some of the design goals for the multi-agent framework discussed throughout this thesis (section 2.4). Then, this chapter presents a discussion on Distributed Artificial Intelligence (DAI) or the study of how Machine Learning (ML) can be applied to Multi-Agent Systems, outlining the possible approaches, and finally deciding on which approach will be used for the learning tasks in Chapter 7 (section 2.5).

2.1 A Taxonomy of Multi-Agent Systems

A significant portion of this overview of Multi-Agent Systems (MAS) and Agent Based Models (ABM) is based on [1] which has excellent and recent survey papers on the very broad and rapidly evolving topics of MAS and ABM. [1] holds that the topics of Multi-Agent Systems and simulations are naturally related. Furthermore, [1] divides up MAS into two categories:

- modeling and simulation for development of real-world MAS (eg. agent-based internet services) and,
- MAS for modeling and simulation of real systems, that is to create artificial in-silico laboratories for modeling complex real-world systems.

This thesis is concerned with the second type of MAS for which a description of possible agent features, agent architectures, agent relations to the environment, agent relations to other agents, and the role of time will be considered in the following subsections.

2.1.1 Agent Features

Below are seven features that agents have in common with Individual Based Modeling (IBM) in the field of Ecology [1].

- Capable of acting with respect to an environment
- Driven by set of tendencies
- Possesses resources of its own
- Partial representation of the environment
- Directly or indirectly can communicate with other agents

- May be able to reproduce
- Autonomous behavior is consequence of perception, representation, and interaction with environment.

In addition to these features, ABM also considers the extent to which agents are intelligent or exhibit learning, in contrast to IBM which primarily deals with purely reactive or stochastic individuals/agents.

2.1.2 Agent Architectures

According to [1] agents employ a reactive, cognitive, hybrid, or open (unspecified) approach to their architectures. Reactive agents maintain basic connections between perception and action without explicit representation of the environment. Cognitive agents reason based on symbolic knowledge of the environment. A hybrid approach attempts to integrate the two previously mentioned architectures, whereas an open approach does not specify a particular architecture. The work undertaken in this thesis does not specify a preferred agent architecture, corresponding to the "open" architecture suggested above. Regardless of specific architecture used, all agent behavior can be described in terms of a "deliberation function" mapping inputs to outputs (actions). Agents are seen as cycling through 3 distinct stages of perception, deliberation, and action during the course of simulation.

2.1.3 Agents and Environment

[1] notes that multi-agent based simulation models situate agents in an explicit environment, which aside from agents, does not exhibit autonomous behavior. However, it may exhibit the following features:

The degree of accessibility of the environment dictates to what extent the modeled Agents can perceive the state of the environment. Localizing agent perception, not only fits in well with the Agent paradigm, it also limits to what extent information needs to be shared between processes in a distributed model, where spatial decomposition is used as a guide for distributing computational load.

The degree of determinism in the environment determines to what extent the state of the environment is influenced by the previous state of the environment . In a purely deterministic environment the state of the environment is entirely determined by its previous state and the actions of the agents. In a purely nondeterministic (or stochastic) environment the state of the environment is entirely random. In practice it may be desirable to model the environment with a variable degree of determinism, so a potential framework may elect to leave this aspect of the environment unspecified. A related concept is the extent to which the environment is itself dynamic, or changing over time, independent of agent actions. Again, because this may vary from model-to-model a framework should not be rigid in this regard.

The degree to which the environment is episodic in nature determines whether episodes, individual cycles of agent perception-deliberation-action, depend on previous cycles. For the purposes of agent based modeling and simulation frameworks, this

decision should be made on the basis of individual models. That is, a likely scenario is agent episodes within a single simulator run should be interdependent, but the episodes in different individual runs should not be interdependent.

Finally, the environment may be discrete or continuous. A discretized environment is divided up into discrete, bounded, areas or regions which the agents may occupy. The most common form of this is a grid-based environment. Conversely, a continuous environment features floating-point distances between agents and elements of the environment. Although a continuous environment implies accurate distances between entities in the environment, one should bear in mind that on digital computer technology all values are ultimately discrete. Floating point numbers incompletely abstract, or obscure, this fact from the programmer. As a result, floating-point errors and artifacts due to behind-the-scenes descretization may creep in to the simulation (depending on the specific implementation of floating point numbers on a particular computer architecture). Another option may be to internally represent the environment in a discrete (albeit potentially finely-grained) fashion, but agent perception can be continuous by virtue of abstraction. This approach works well with the method of using spatial decomposition as a guide for distributed processing, mentioned above.

2.1.4 Agent Interactions

In general, [1] indicates that, agents may communicate explicitly by passing messages, or implicitly using signals. Messaging is explicit because the sender assumes that the recipient(s) will understand the message. Signals, on the other hand, are tokens left by agents as they traverse the environment, and can be implicitly interpreted by other agents

which perceive these signals. Marks are discrete signals, whereas fields can convey continuous values which may differ in different parts of the environment. Fields can communicate complicated information such as attraction or repulsive forces. Support for both explicit message passing and signals should be considered.

2.1.5 Time

[1] raises several issues related to the modeling of time. The first is continuous time, which requires continuous time functions which determine the state of the system for any possible instant in time. This approach is rarely used. There is discrete time in which the system always advances by a fixed, discrete, time step. There is also discrete event based, where time advances discretely between (usually scheduled) events, however the interval of time between events can be a real or continuous value.

In a simulated MAS, there are several aspects of time which must be considered. The most obvious of these is modeling the timed cycles of agent perception-deliberation-action. With a few exceptions [2] simulated agent perception-deliberation-action is instantaneous for the sake of simplicity. Due to the difficulty of implementing continuous time functions in complex systems such as MAS, most approaches tend to opt for discrete time or discrete event-based time models. In models where there is a spatial aspect and agents may exhibit reactive behavior such as micro-social distancing, agents will ideally react “as often as possible” in practice resulting in a quasi-discrete time step (however agents may be slightly out of step owing to the real valued timestamp of events). As with the discussion on discrete versus continuous state, on digital computers, a continuous or real number is in reality discrete (although highly accurate), this fact is

abstracted away from the programmer. Similar results may be achieved with a finely grained discrete time step, with some agents "sleeping" through time steps for efficiency if it is known the agent will have nothing to do on a particular time step. In any case it should be noted that a discrete time scheduled system can emulate a discrete event simulator if the discrete time system is capable of representing real-valued times, as well as being able to adjust the time step. A time step can be chosen based on the activation time of the earliest next "event". Similarly a discrete event system can be made to emulate a discrete time system simply by generating a periodic "heartbeat" event with a fixed time interval. The heartbeat event would activate all agents when it is processed.

Second, if the environment is of a dynamic nature (as previously discussed), the temporal aspect of the evolution of the environment must also be considered. [1] suggests that the temporal evolution of the environment be coupled with the means of advancing the agents (usually discrete time, or discrete event based) for simplicity.

Finally, [1] considers the modeling of the temporal evolution of the system due to agent actions. In other words agent actions need to be coupled with the evolution of the environment (including other agents). One typical approach to simulating a discrete time system is to simulate each agent 's time step (where the agent performs perception-deliberation-action) in turn, after which the time step of the environment or entire simulation is advanced by the time step, and the cycle begins anew. This is potentially problematic, because a different ordering of agents will result in a different final system state, even though agent behaviors have not changed, only the order in which agent actions are applied are changed. This can be solved by shuffling the order in which the agents are advanced, or by fixing the system state until all agents have acted, and the time

step of the environment has been advanced. Still, if the agents take concurrent actions, some mechanism (usually non-trivial) must be implemented to resolve conflicts arising from these concurrent actions. This is a problem inherent to both discrete time, and discrete event simulations. This is exacerbated by the "unrealistic" [3] identity of agent action and change in the environment or other agents. One solution is to decouple agent action and environment evolution through message passing, which also conveniently abstracts away from the perspective of the agents the fact that interacting agents may in fact be in distributed memory spaces. [1] notes that, as with discrete time based systems, there are related ways of dealing with conflicts arising from concurrent actions in discrete event, or scheduled, systems.

2.2 Strengths and Weaknesses of ABM

The strengths of ABM are made in-depth throughout this thesis. In the context of the current discussion it suffices to list strengths including: complex emergent phenomena may be explored; such models can be as detailed as one can afford to make them - or as detailed as available data supports; ABM can be used to investigate "what-if" scenarios; and the close resemblance of the resultant ABM to the natural language description of the source system makes ABM an excellent communication tool between modelers and practitioners, especially when leveraged using visualization technology. However, there are several weaknesses to the ABM approach, mostly due to the same factors to which ABM owes its strengths; that is that the ABM itself is a complex adaptive system. These weaknesses or drawbacks will be discussed in the remainder of this section. Then the

discussion will turns to means of addressing model validation and questions regarding correctness of the model and simulator. Finally means of addressing the question of replication by other researchers are presented.

2.2.1 Criticism of ABM and some challenges

Due to the complex adaptive nature of ABM these models are well known to exhibit high output variance and sensitivity to initial conditions, further evidenced throughout the rest of this thesis. Sensitivity to small variation initial conditions makes results difficult to replicate. Because of high variance and sensitivity to initial conditions, a large number of runs are often needed to estimate "average" or "expected" behavior of the modeled system. Therefore, a simulation study using ABM is often very computationally intensive. ABMs typically feature a large parameter space to explore due to complexity and the high level of detail in these models. The combination of the aforementioned factors make ABM validation difficult or perhaps intractable in the traditional equation-based modeling sense.

Because very detailed models are possible, modelers have to be careful not to "infer" too much or introduce data that cannot be statistically validated. Data collection and availability for both choosing input parameters and model validation has a long way to catch up to modeling capability as outlined in Chapter 4. As discussed in Chapter 6, ABMs are often used for investigating "what-if" scenarios, which attempt to probe situations which have not occurred in reality. There is by definition no data to validate events that have not happened.

ABM does not provide a succinct mathematical or symbolic description of the modeled system as Queueing (discussed in Chapter 5) or other equation based models do. However it may be possible to use an ABM to induce a mathematical or symbolic expression describing the modeled phenomena, using Genetic Programming as described in section 7.2.1.

2.2.2 Validation and Verification of ABM

Validation and Verification in the field of ABM are a topics of intense research. [1] reports that The Zeigler Framework for Modeling and Simulation [4] identifies 3 types of validity to consider with respect to an Agent Based Model running on a Simulator. The first is replicative validity, which considers the extent to which behavior exhibited by the model is observed in the source system being modeled. Second, predictive validity implies replicative validity, but in addition considers the extent to which the model is able to predict behavior of the source system that has not yet been observed. The third and strongest form of validity is structural validity which requires predictive validity, and furthermore requires that the model matches the way in which all components of the source system change state with respect to input.

The Zeigler framework also identifies 3 questions a modeler should ask with vis-à-vis the validity of the model. The first of these questions is to what extent does the model represent the source system, in terms of the 3 levels of validity mentioned above? The second is to what extent does the model accommodate the suite of experiments to be performed using the model? This question is highly domain specific. The third question is to what extent does the simulator correctly implement the model? The third question has

to do with the nature of the simulator, specifically factors related to reproducibility which will be addressed in section 2.2.3. In other words, [1] identifies a distinction between model validity and simulator validity, both addressed separately below.

Empirical validation involves assessing to what extent the model represents the unknown processes that led to some observed data in the target (real-world) system being modeled. However, with respect to ABM in particular this is still an open research problem [5] and there is little consensus [6] on precisely how this should be performed, perhaps owing to the high variance in results, and large parameter space these models tend to have. It is also advisable to validate the agent interactions (micro-level) separately from the emergent behavior (macro-level) in the model [5], which is the approach evidenced by EpiSims, and BioWar, below.

Model alignment [7] also known as docking, is the process of setting up as identical as possible scenarios in two independently created models of the same system to see if the results are the same or statistically indistinguishable. Actually, this approach is a form of the following plausibly reasoned argument [10]: It would be very unexpected if two unlike models will give statistically indistinguishable output given the same input. Therefore if the models give statistically indistinguishable output given the same input it strongly adds credibility to the statement that the models produce the same results. If one of the models is well established and believed to be accurate within a particular field, this is taken to be evidence for the validity of the second, less established model. Since this is the “best practice” in the field of ABM it is often accepted as “proof” of validity, however it is likely insufficient “proof” for some critics.

In this context, it is important to “Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful” -Box and Draper, (Empirical Model-Building, p. 74); a quote from one of the most influential texts on engineering modeling. Furthermore, computer scientist/mathematician/physicist Stephen Wolfram has been quoted saying: "I do think that the history of the universe... (is computationally) irreducible--so we actually have to live it in order to see what happens." This quote suggests no model can hope to reliably predict the future, present, or past for that matter. In fact, there is nothing to say that what events and processes happened to generate real-world data resulted in most likely outcome given some imperfect description or assumptions of the initial state of the world. For example, the fact that there is life on earth is thought to be tremendously unlikely given all possible outcomes of “random processes” from the beginning of universe until now. These issues with modeling aren’t limited to ABM; mathematical models are similarly challenged, as alluded to by Paul Dirac: “This result is too beautiful to be false; it is more important to have beauty in one's equations than to have them fit experiment. “

The standard means of demonstrating ABM validity in a well established field such as epidemiology as evidenced by the case of EpiSimS [8][9] is a form of “plausibly reasoned” argument. Essentially, over the course of time, model predictions that are true add to the credibility of the model moreover if the prediction is counter-intuitive, improbable, or unexpected [10]. EpiSimS is a widely used simulator and related disease model developed by the United States, Department of Energy (DOE) at Los Alamos National Labs, supports confidence in the model by validating individual model components (transport model) and then appealing to the utility of the model (as

evidenced by a number of peer reviewed publications) rather than establishing overall validity of the model/simulator in a rigorous mathematical manner or "proof" of correctness. This is the de-facto standard of proof in MAS as they apply to science owing to their high variance and complex nature. A similar validation route is taken by BioWar [11], validating its climate and social network models separately.

Again we are reminded of the difficulty of validating events that have never occurred. Can such things be modeled? Even if there are observations of what did happen in a human society the outcome is one of many possible outcomes since society is itself a complex adaptive system. What happened was not necessarily the most likely outcome. Clearly, this limitation of ABM is related to the human limitation of imperfect knowledge of things that are, and no knowledge of things that are not.

In the end, [1] concludes that validating model behavior in social domains is highly subjective, and encourages real stakeholder and expert participation in the modeling process, including validation. Experts are more likely to have given hypothetical scenarios some thought, and by observing model visualization may be able to point out implausible features, assumptions or outcomes. Furthermore, [1] asserts that initial goals should include achieving replicative validity, and notes that ultimately Virtual Reality (VR) technologies should be leveraged to achieve stakeholder or expert participatory design of the model. This supports the focus on visualization throughout the design process of the modeling framework developed in this thesis.

"Validation and calibration of the model through expert judgment is crucial. ABM is often the most appropriate way of describing what is actually happening in the real world,

and the experts can easily 'connect' to the model and have a feeling of 'ownership.' " [12].

2.2.3 Issues surrounding simulator verification

The nature of the simulator on which the model is built is fundamental to the question of reproducibility or replication of the model by other researchers. Ultimately the more widely used simulator, with more people being exposed to and testing its code is a pragmatic and possibly the best assurance of simulator validity and verification.

[1] identifies several categories of simulator. First are "one-shot" simulators, which represent the most domain-specific, even experiment-specific, simulators. Very limited numbers or classes of model can be run on this type of simulator. Second, "domain related" are simulators capable of executing multiple models in a particular domain, for example infectious disease spread. Normally simulators of this type will have support for a number of domain features built-in such as a spatial model (or not), while they will remain flexible enough to implement many different models from the domain. The third type of simulator is the "generic" kind, sometimes referred to as "frameworks", because they often specify the simulator architecture while leaving any domain specific features up to the programmer.

Generic frameworks are the most re-usable type since they can be potentially applied to any domain. They are the most widely used, and therefore regarded as the most widely tested and vetted simulators. However, their use presents a new set of challenges, which will be discussed in more detail (along with one-shot and domain related simulators) in section 2.3. [1] suggests that the existence of far more one-shot simulators than the other

types is because there is no consensus on the way simulators aught to be implemented and what features they should have. Furthermore, [1] hypothesizes a "gap" between the MAS paradigm and available simulation frameworks. Section 2.4 identifies some of these gaps, and proposes a set of desirable features in order to "fill the gap".

In contrast, a one-shot simulator is comparatively easy to implement, and gives the modeler fine control over the simulator processes, enabling them to fulfill their requirements. Typically, in order to minimize development effort, the designer will make assumptions which ease the implementation of the model at hand, without consideration for how these assumptions will constrain or complicate re-purposing the simulator to implement a different model. From a software engineering perspective, part of the reason that one-shot models are so easy to produce is that little or no effort go into making the software reusable or extendible. The large number of one-shot simulators is problematic because by their nature they are difficult to re-use. The reusability of the simulator in turn affects the reliability of the simulator; the more researchers that (re)use a particular simulator the more chances that bugs will be identified and fixed.

Turning back to the topic of reproducibility by others, using a widely-used simulator framework makes it easier for other researchers to reproduce results. It has been shown that the results of simulations are highly dependent on implementation details. Utilizing a common back-end implementation and identical design choices with respect to simulator characteristics promotes reproducibility. Reproducible results tend to be regarded as more plausible or reliable. [1] summarizes that simulation results are often published without enough detail about the model or implementation in order for other researchers to be able

to reproduce the results. Therefore, [1] suggests that the field of ABM and MAS aught to promote re-use of existing work, as well as to define unambiguous models.

Towards the re-use of models and simulation frameworks, one suggestion is to use an existing, well-vetted, and popular MAS framework such as Swarm [13], especially since it is open-source already. However, as will be demonstrated in section 2.3.4, Swarm comes with its own set of challenges, mostly owing to its age. No choice of framework will satisfy all modelers, as evidenced by the lack of consensus on which framework is the best choice.

Towards the definition of unambiguous models, the ODD (Overview, Design concepts, Details) protocol has been proposed [14]. By following the ODD protocol, model documentation will naturally address the following questions:

- What is the purpose of the model?
- What agents are being modeled? What are the attributes or state variables of the agents? What are the spatial and temporal units and boundaries of the model?
- Which agents perform what actions, and in what order? When is agent state updated. How is time modeled?
- What is the initial state of the model?
- What scientific theories underlie the model design?
- What are the model results that are emergent from agent interaction?
- What are the agent's adaptive traits?
- What are the agent objectives that affect and are affected by agent adaptation, if any?

- What form of learning do agents exhibit?
- What predictive abilities to agents have?
- What sensing capabilities do agents have?
- What are the types of interactions between agents?
- What processes in the model are (pseudo)random?
- Are agents aggregated into any kind of hierarchies?
- What data or output is collected from the model?
- What is the nature of input data? Data files as opposed to the output of other models?

The ODD protocol is certainly a step in the right direction especially where the simulator engine is proprietary or otherwise closed-source and is possibly sufficient for simple agent based models and simulators, and necessary in the case of one-shot simulators unlikely to be gain widespread recognition or use. However, towards the goal of documenting an ABM for replicability it does not go far enough, since even switching a single “less than” operation within a simulator statement to a “less than or equal to” can make a dramatic difference. Due to high output variance and sensitivity to initial conditions it is still a research question as to the criteria for declaring the output of two models (or even the same model) to be the "same" or "successfully reproduced". This suggests that the essence of the model is not further meaningfully reducible further than the source code.

A conjecture of this thesis is that the best way of documenting the ABM is sharing the code in an "open-source" development model. A successful open source project is one where many researchers would use, read, write tests for, extend, and improve the

simulator framework code. Other researchers will publish results using the open source project, further adding to a database of validation for the simulator framework.

As far as specific tests for simulator verification, the same general principles of Software Engineering apply, for example unit testing and code review [15]. Other approaches specific to ABM are suggested by [5]. The first of these is checking whether the simulation continues to operate sensibly for extreme values, or boundary conditions of input parameters. Another is to perform a sensitivity analysis to verify whether the simulator performs sensibly in response to changes to input parameters or combinations of these parameters. The third is a destructive testing method based on non-linear testing [16] which uses an automated optimization method (such as Genetic Algorithm) to search through a "reasonable" parameter space for combinations which result in "unreasonable" model output. [5] concludes that a simpler simulation is preferable to a more realistic one in terms of verification and validation, corresponding to a reduction in the parameter space that has to be investigated.

2.3 Survey of Agent-Based Models and Simulators

The work presented in this thesis, is likely the first domain specific ABM for infectious disease spread in the Emergency Department, however it is discussed in the context of an ABM framework being built up for spatially-explicit, human-centric, human-scale models. To consider the merits of the proposed framework, it is forthcoming to discuss the features, and drawbacks of existing multi-agent frameworks for simulation. This

discussion will focus on healthcare related ABM, as that is closely related to the central theme of this thesis, however examples are presented from other domains where appropriate (ex. robotic soccer).

2.3.1 One Shot

Originally many single-purpose or "one-shot" ABM models for social science studies including both spatially-explicit (segregation) [17] and more abstract (prisoner's dilemma) [18]. These sort of one-shot simulators were among the first ABMs in social science, as well as the field of ecology where the technique was referred to as Individual Based Modeling (IBM) [19]. For brevity, this chapter will not exhaustively cover the multitude of one-shot simulators developed for various domains, as these models are of limited interest because of barriers to their re-use as discussed in section 2.2.3. Three examples of one-shot models include:

- DSSW [20] is an agent based model of disease spread in Winnipeg, Manitoba. Agents move about the city based on schedules of going to work, school, shopping, etc. This model is written in Microsoft Visual C++, runs on a desktop PC, and features real-time visualization of model operation.
- From the area of economics, there is an agent based model for simulating an artificial power market for planning within the power generation industry [21]. The model considers residential consumers using power to heat their homes. A number of indicators are presented to the user in graph form.
- All robotic soccer simulations are by definition agent based models. The most widely used robot soccer simulations is the Simulation League platform provided

by the Robocup Federation [22]. This particular software is spatially oriented, supporting 2D and 3D environments, as well as featuring visualization capabilities. The source code is available at http://sourceforge.net/projects/sserver/files/rcssserver/

2.3.2 Domain Specific

In time some single purpose ABMs are developed further into more general models, a process called refactoring [15]. Sometimes these more flexible simulators are created with some reuse in mind.

An early example of a domain-specific, somewhat generalized, simulator framework for social science is Sugarscape [23]. Sugarscape features a spatial environment with visualization in the form of a cellular-automata type discrete grid. Time advances in cycles which appear to be equivalent to arbitrary time steps. Communication between agents is achieved in the form of "tags" which can be exchanged between agents. The environment may or may not contain "food" or other non-agent entities which have some utility to the agents. Sugarscape has been used to model such social phenomena as emergence of free markets, war, famine, and infectious disease spread. Sugarscape appears to be targeted towards the desktop with little consideration for running in parallel or distributed mode. There are multiple implementations of Sugarscape, some of which have source code available [23], but regardless provide a common implementation platform for social simulation.

BioWar is a domain specific ABM for simulating bioterrorist attacks that "combines computational models of social networks, communication media, and disease

transmission with demographically resolved agent models, urban spatial models, weather models, and a diagnostic error model" [11]. It uses a number of publicly available data sources such as census tracts, school district boundaries in an ad-hoc manner. Like many ABMs it can be used to address "what-if" questions relating to public health scenarios. It employs the concept of location on an urban scale (businesses, schools, parks), but travel between locations is treated as abstract. The simulation is time stepped with a time step of four hours. BioWar is interesting in that agents exhibit some form of learning or reaction to bioterrorist attack. [11] reports no online visualization or GUI capability, and notes that BioWar is parallelizable at the individual run level, that is many runs can be run concurrently in a Monte Carlo type investigation.

EpiSimS [9] is an event based Agent Based Simulator for modeling infectious disease spread through a single urban environment of up to 1,000,000 agents. It has been developed at Los Alamos National Laboratory as extensions of the TRANSIMS model for transportation systems. As a result, the simulator is capable of importing transportation systems data made available by the United States Department of Transportation (USDOT). The simulation is spatially divided into locations (city blocks) which are further divided into sub-locations (buildings). USDOT data on movement of persons, previously used for transportation systems analysis and planning via TRANSIMS, is used as the basis for contact between agents in EpiSimS. EpiSimS has a distributed mode which uses MPI [24] to achieve parallel model execution. It also has limited visualization capabilities to view model results after the run is completed, but it is unclear whether there is some mode of observing the simulation while it is running for

debugging and validation purposes. The source code is available to universities at the discretion of Los Alamos Labs and for a few of $1000 USD.

CROWD [25] is a simulator for human disease propagation in small urban environments, intended to be run on a desktop PC. It is a modified detailed, spatial, urban-scale combat simulator used by various commonwealth defense agencies. Due to the military origin of this simulator, only a limited amount of details are available regarding its implementation. [25] makes it clear that demographic and actual spatial data were used in formulating models, however it is unclear to what extent this process is automated to use standardized formats such as GIS. Although model output is visualized by using circles of size corresponding to the number of infected individuals overlaid on a city map, it us unclear as to whether any visual or graphical interface is available during a particular run for visual debugging and validation. [25] does not reveal much about the spatial aspect of the model other than saying that it is more detailed than that of BioWar and EpiSimS. Due to the proprietary nature of this simulator, it is difficult to extend, reuse, or independently validate model results.

HealthSim [26] is a framework for discrete event simulation of health care delivery in varied institutional settings developed at Argonne National Laboratory. HealthSim has an extensive models of human physiological processes, cognitive behaviors, clinical processes, and medical equipment. It has some support for running in distributed mode. It provides as output a variety of reports regarding patient flow and financial aspects.

2.3.3 Generic Frameworks

Through successive phases of refactoring, ABMs that become generalized to a point where many varied models can be made using them, and therefore can be considered frameworks. In other instances frameworks are designed and built from the ground up to be general purpose. [1] provides a broad survey of existing frameworks which could be used to construct ABMs.

Recently, the computing trend of Moore's Law has been manifesting as an exponential increase in the number of operations that can be performed concurrently, rather than the historic increase in the number of operations that can be sequentially executed (corresponding to increases in the number of CPUs or cores rather than increases in clock speed). Therefore software, including frameworks for MAS should take concurrency and distributed computing into account.

The High Level Architecture (HLA) [27] has been proposed by the U.S. Department of Defence as an interface specification facilitating communication between disparate simulation engines, perhaps written in different languages, and running on different machines. The HLA specifies an interface for a "Runtime Environment", but due to its high level nature the HLA does not appear to offer specific solutions or optimizations for efficiently distributing computation between multiple CPUs. Therefore, each simulation suite based on HLA is forced to solve the problem of efficient computational distribution anew, possibly leading to scalability concerns for the entire system. Scalability is without doubt a pressing issue to the usability of such frameworks, and this potentially difficult

problem should not be left up to the modeler. This is possibly due to the conceptualization around using a grid rather than cluster environment.

The Latent Energy Environment (LEE) [28] toolkit is a general framework for research on complex adaptive systems. Despite this broad mandate, LEE imposes quite a few design choices or constraints affecting the variety of models that can be implemented using LEE. The agents have an artificial neural network (ANN) architecture which is used to specify the behavior of agents. Along with sensor inputs to the ANN, internal state is used to determine agent actions. The environment is specified to have a 2 or 3 dimensional toroidal grid topology. Similar to Sugarscape, LEE has "food" in the form of "energy" which may be present in the environment, that may be consumed by agents and may be replenished over time. This framework is time-stepped with each agent being probabilistically executed at each time step. Virtual concurrency is achieved in this way, but LEE does not appear to have support for truly distributed computing environments. LEE features some integrated visualization capabilities.

The James II framework [29] is designed to implement general-purpose modeling and simulation systems, but has been somewhat tailored to the needs of ABMs. It supports a variety of formal agent architectures. A spatial aspect to the environment is not specified, and this framework features a simple GUI for model construction and visualization. James II uses a (parallel) discrete event formalism called (P)DEVS. A master server automatically partitions the simulation across processors when running in distributed mode. It is not clear whether this partitioning can take advantage of natural spatial hierarchies in human-centric systems (ex. people in a room or institution will tend to primarily interact (and share other commonalities) with others in the same room or

institution) due to the unspecified nature of spatial environments in James II. Communication between model partitions is done in a hierarchical manner resembling a tree. [1] notes some concerns about PDEVS based simulations in this context. The first is that events may only be executed in parallel if they occur at precisely the same timestamp. The second is a potential for a communication bottleneck perhaps due to a naïve load-balancing algorithm. This framework includes some limited capabilities for visualization. Source code is available on the project website [29].

SeSAm is a multi-agent simulation framework with a well developed GUI for configuring models as well as visualizing results apparently targeted towards a desktop platform. Both 2D and 3D grid-based environments are included, and supports both discrete event and discrete time stepped execution. Source code is also available at the project website [30].

Swarm is an open-source framework [13] for general multi-agent systems dating back to the 1990s. It is discrete-event based and does not specify a spatial model for the environment, however a 2D grid environment is available via plug-in, - complete with GIS integration support. GUI and visualization components are available via plug-in. Swarm has no standardized support for distributed processing. Researchers implementing their own means of distributed execution, may find that these efforts are significant when compared to the framework facilities offered by Swarm, and may even significantly impact the reproducibility of the resulting models. Nevertheless, Swarm has been used as the basis for at least one vector-borne disease model [31].

The RePast framework for agent simulation for which offers the source code under an “open source” license [32], and is sometimes considered a modern successor to the aging

Swarm framework as the two share a number of conceptual similarities. RePast, like Swarm is discrete event driven and supports virtual concurrency by randomly interleaving concurrent events. A number of features support the construction of human-centric models such as 2D and 3D grids. This framework also includes facilities which allow it to be partially data driven, in that Geographic Information Systems (GIS) can be used as input when modeling a real-world environment. Although these features do not make it overly-specific, RePast reinforces its flexibility by the core simulator code being agnostic to the specific structure and behavior of models. RePast appears to provide a GUI with visualization and limited simulation control and configuration capabilities. RePast has been adapted for use with the HLA mentioned earlier, called HLA_RePast, to achieve parallelism and distributed processing. HLA_RePast introduces the notion of exclusive variables to resolve conflicts, and effectively emulates the functionality of a mutex across the distributed architecture. Despite conservative synchronization, to eliminate the need for rollback or conflict resolution, [1] concludes that communication overhead may become a bottleneck in some cases. This is possibly due to the rather abstract nature of distributed simulation offered by HLA, without any specific provisions for distributed communication efficiency.

SIM_AGENT [33] is a general agent framework written in the obscure Pop11 language, meant to model a variety of agent architectures from physical agents situated in a spatial environment to abstract or Internet-based agents. The simulator time-stepped, with agent perception, deliberation ,and action happening on every time step. There is a parallel or distributed version which uses HLA, called HLA_AGENT. When running in the distributed mode, agents being simulated in one process may need to interact with

those in another process. In this case, proxy agent is created in the remote process, and its state is updated with that of the original agent being simulated elsewhere. As with HLA_RePast, [1] notes that communication may be a bottleneck in some cases. Possible inefficiencies arise when the entire environment is simulated in a single process, and when naïve partitioning is used; since partitioning has to be done in an ad-hoc manner for each new model.

CHARON is an agent framework [34] which uses a specialized language for specifying agent architectures and hierarchies. The CHARON agent specification is compiled and run on the associated simulator. The CHARON framework has notions of both discrete time steps and continuous variables. [1] notes that the complex discrete/continuous hybrid approach of CHARON has created some challenges in terms of parallelizing the framework for distributed computing environments. A "conservative" mode of synchronization where each distributed process completes one time interval in lockstep with the other distributed processes. The performance of this mode of distributed operation has been found to be quite poor, in practice, due to communication overhead. A second mode of distributed computing attempts to predict the input from the other distributed processes. When prediction fails, a rollback of the simulation state is sometimes required. Nevertheless, this approach is seen as having the potential to usefully speed up CHARON simulations.

Gensim, and its distributed version DGensim are simulators written in Allegro Common Lisp, and targeted, as the name suggests, for simulating generic multi-agent systems. For this reason, no explicit environment models or agent architectures are implied. DGensim is very interesting for the fact that it focuses on real-time agent

performance/simulation, in that agents have a wall-time based time-slice in which to perform their sensing, deliberation, and action. Given n processors, agents are partitioned across n-1 processors with the remaining 1 processor simulating the environment via the Environment Manager. Each processor which is executing agents also runs an Agent Manager to interleave the execution of each agent on that processor. Agent decisions are timestamped and sent to a central Environment Manager which sorts the decisions by timestamp and applies them sequentially to the environment. In general, the single processor devoted to simulating the environment may be seen as a bottleneck (especially if there are a large number of agents) since some local interactions between agents could be handled on the n-1 agent processors, locally on that processor, provided that the agents being modeled exhibit bias towards interacting with local agents, and that the locality property is exploited when agents are partitioned across processors. The challenge of maintaining synchronicity within a distributed real-time model is approached in three ways by the authors of DGensim [2] who prefer to avoid rollback. If an agent action is delayed, for example over a network socket, such that it is not received in time by the Environment Manager before the global time step is incremented the action may be ignored as if it never occurred. The second choice is to process the action as if it occurred on the time step during it was received, rather than when the action was generated. The third approach implements a lock-step mechanism where the Environment Manager expects to receive an action from each agent before the global time step is advanced (even if that action is to do nothing). Although developed for a cluster environment in the 1990s, this architecture is worth investigating on modern multi-core desktop machines.

MASON is a general-purpose discrete-event simulator written in the Java language enabling reproducibility across platforms [35]. It was originally developed to model a relatively small number of robotic agents, so it has been designed with multi-threaded desktop models in mind, rather than cluster-based computing. In practice this limits "efficient" simulations to about 1,000,000 agents without visualization. The visualization capabilities, are quite impressive when used, however. Modules are available for importing social network data from a common format (JUNG). Other modules include a 2D physics module as well as evolutionary algorithm toolkit. The source code is available for download from the website (http://cs.gmu.edu/~eclab/projects/mason/). MASON supports 2D or 3D, continuous or discrete topologies in square, hexagonal, or toroidal configurations. It also supports network/graph based environments.

NetLogo [36], StarLogo [37], and Breve [38] are three independently created simulators and tools for the common goal of teaching and rapid prototyping artificial life systems. StarLogo and NetLogo feature 2D grid based environments, whereas Breve focuses on 3D continuous environments. All three have extensive visualization features. The source code for StarLogo and Breve are available, whereas the source code for NetLogo is not.

SPADES is a framework [39] for implementing distributed discrete event simulations with a relatively small number of compute intensive agents. Specifically, it has been tailored to the needs of simulating robot soccer teams. Hence, a two dimensional spatial world model is assumed, complete with soccer ball. Each process which contains agents has a communication server which manages updates from the central world model and simulation engine which run together on a single processor. Once agents process

environmental information and decide on a course of action, the action is sent to the central world model through the communication server. [1] notes that this approach incurs considerable communication overhead. These scalability issues are possibly acceptable for the small number of agents for which the framework was designed. A great number MAS geared towards robot soccer applications are in existence owing to the yearly RoboCup [22] competition. We will not attempt to review them all here.

Anylogic is a closed-source, proprietary, and commercial simulator engine and related libraries for developing multi-paradigm models including agent-based modeling [40]. Anylogic includes a means of visually or graphically programming agents through statecharts [41], and action charts. Several libraries are provided with ABM in mind such as a pedestrian library for modeling human movement along with an associated spatial model. There is also considerable support for visualizing models. The developers of Anylogic do not appear to officially support HLA or any other means of distributed simulation. As a result, Anylogic may be suitable for desktop simulations with a relatively small number of agents. To summarize, although Anylogic has many advanced features, it is closed-source, proprietary, and commercial. This leaves no way for the modeler or researcher for verification or validation of the simulator other than trusting the manufacturer. This may be acceptable for business applications but is clearly undesirable for research and scientific applications, which potentially include public health.

2.4 Design Considerations for the Framework Presented Herein

Clearly one-shot ABMs and many frameworks we regard as too specific to be easily extended to other modeling domains. However, the most general and extendible or flexible frameworks at the same time provide very little structure as well as facilities (behavior and support modules) to model specific human-centric agent features – in other words not very practical. It is an assertion of this thesis that a goal of future undertakings in Agent Based Modeling or Simulation frameworks should be conveniently extendible to modeling human-centric domains, in other words a "sweet-spot" between flexibility, extendibility, and specific support for human-centric domains. The framework should, as a consequence, be extendible to interface with Machine Learning modules. It should also support facilities for incorporating real-time data, and visualization as this has been noted to be an essential component of ABM, in the roles of communication, model validation and development. This visualization tool can be extended to also serve as a tool for model environment construction or editing environments imported from data. The accessibility of agent behavior development to persons with a non-programming background can be improved by first providing a scripting layer on top of the compiled code, and then perhaps adding a visual programming (drag and drop) on top of that. As mentioned, the rise of parallel or distributed computing systems also suggests that contemporary or future agent based simulator frameworks have support for distributed, parallel, or cluster computing. The increasing availability of cluster-based compute resources (a

consequence of Moore's Law), sensitivity to real-time computational constraints, and medical data privacy issues auger for cluster-based computing and against crowdsourced ad-hoc peer-to-peer distributed compute resources popularized by projects such as SETI@Home [42]. Given these disparate goals, some tradeoffs will likely have to be made between features, based on feature utility. Again, [1] considers the great number of one-shot simulators in existence to be evidence that this "sweet-spot" has not yet been found.

To summarize, the ABM application here is specific to hospital emergency departments, yet is presented in the context of the simulator framework being developed around it with several broad design goals. These goals include:

- Human-centric modules/extensions/features/support (Chapters 3,4,5,6). This includes idea of being spatially oriented since humans are physical entities that occupy and traverse space rather than consisting of only information. The use of human time scales on the order of seconds is also suggested.
- Scalability in terms of distributed processing (Chapter 7). Heuristics for efficient (re)partitioning of the simulation across distributed processes should take into account human-centric heuristics for (i.e. spatial hierarchy orientation). According to [1] whether or not partially observable environments will lead to conditions in which a distributed model will experience speedup is an open research question, and beyond the scope of this thesis.
- Capability for data-driven operation (Chapters 3,4, and 5).

- Visualization for communication, model development, and validation (Chapters 3,4,5,6, and 7).
- Extendibility and re-use (Chapters 3,4,5,6 demonstrate these features). Extendibility in order to interface with Machine Learning (ML) systems is the focus of Chapter 7.
- Reproducibility by others (Chapters 3,4,5, and 6) – The advocated solution is to open-source the framework and models once refactoring is complete.

2.4.1 Applicability to other problem domains

The framework described in this section is also applicable to many other problem domains with a spatial aspect and which feature some kind of transfer between agents, for example:

- War games
- Self-organizing logistics systems for supply chain management
- Social models of panic spread through a crowd
- Transportation models for urban planning
- Models for economics surrounding tangible goods.
- Coordination problems involving multiple robots.
- Other complex systems where analytic methods may be intractable

In all these cases the potential for Machine Learning in association to the ABM is implied.

2.5 Approaches to Machine Learning in Multi-Agent Systems

Before discussing Machine Learning (ML) approaches specific to MAS, also known as the field of Distributed Artificial Intelligence, DAI, some terminology will be introduced in order to frame the discussion. For a more in-depth introduction to ML, [43] is an excellent reference.

2.5.1 Introduction to DAI

Firstly, it serves to distinguish between online and offline learning approaches or algorithms. Online approaches learn in an incremental fashion, being exposed to one training example, instance, or episode at a time. The online approach learns continuously as new training instances are presented. In contrast, an offline approach presents all training data or samples at once, and no learning occurs outside of this training period.

Another means to categorize ML approaches is in terms of the information content of training examples. In a supervised learning task, the learner is provided with the correct target or output values along with each set of input values. That is, the learner is provided with the true label of each example in the set of training examples. In a reinforcement learning task, the true label is not provided, only a reward signal indicating how well the learner is performing the task. In an unsupervised learning task there is no feedback signal whatsoever, making these tasks suitable for clustering operations, rather than learning a target policy.

A third means which can be used to compare Machine Learning approaches is the concept of "A to I ratio" (Artificial-to-Intelligence ratio) as outlined in [44] to mean the ratio of what is delivered by the automated, artificial method over the amount of human intelligence being applied to the problem. Therefore approaches with a low A to I ratio have a considerable amount of human intelligence supplied in order to solve a problem. Conversely approaches with a high A to I ratio exhibit a relatively low amount of human problem solving to the value added by the artificial operation of the algorithm. [44] argues that a high A to I ratio is preferable, because in the case of the alternative it is really humans solving a problem with machines merely "crunching the numbers", in other words not exhibiting much intelligence at all.

Other aspects of ML approaches include human readability factors. It is desirable for the output or result of learning to be interpretable by humans for validation purposes. The results may also be generalized to add constructively to the body of human knowledge through the process of induction. Clearly, symbolic approaches have an edge over numerical approaches in human readability by virtue of its symbolic approach. Also, there is the issue of the scalability of approaches to distributed learning environments. Section 2.5.2 will discuss various approaches prevalent in DAI literature, with respect to the first three means of categorizing DAI approaches introduced in this section. The significance of how easily an algorithm can be parallelized will become apparent in a section 2.5.3.

2.5.2 Survey of Approaches

In the literature, most often, ML approaches applied to MAS fall into one of the following categories: methods specific to reinforcement learning problems [45][46][47][48][49][50], evolutionary approaches [45][47][51][52][53], game theoretic methods [46], Artificial neural networks (ANN) [52][54][55], imitation learning [48][54][56], and custom one-off approaches [57][58] .

The formulation for Reinforcement Learning (RL) approaches is closely tied to the concept of agents enacting a policy (sensing, deliberation, action) with respect to the state of the environment [59], and has a strong emphasis on online performance in the form of a reinforcement signal from the environment. Due to this natural affinity, many DAI applications utilize RL algorithms at their core [48][49][50]. RL methods typically feature high A to I ratios. One exception is human assisted analysis of the problem with respect to definition of continuous input variables or ones with a wide range of discrete values. Such analysis is necessary to prevent the definition of an overly large number of states. The result of learning is a state-action table containing accumulated rewards, and is not particularly amenable to human interpretation. There have been some recent efforts to parallelize RL methods, with promising results, however distributed RL methods are still in their relative infancy [60].

Evolutionary approaches including Genetic Algorithm (GA) and Genetic Programming (GP) are a flexible algorithms that have had many successes in online [51][53], offline [61][62], supervised [61][62], and reinforcement learning [47] tasks. These algorithms are general purpose, biologically inspired, means of optimization. They

employ “natural selection” and “heredity” to improve a population of solutions between successive generations. GA produces a parameterized numerical solution. However GP produces a symbolic representation (computer program) potentially allowing humans to interpret the result of learning, as a step in the process of induction for knowledge acquisition. GP has well documented scalability for parallel execution and among the highest A to I ratio of all known ML approaches [44]. There are numerous published applications of evolutionary approaches to DAI problems including [47][51][53].

Game Theoretic methods [46] have been successful in problem domains where AI rivals the best human performance, for example two-player limit Texas hold’em [63] and Chess [64]. These methods are offline and supervised and involve abstracting and parameterizing interactions between the agent and its environment (usually other agents) into utility or payoff tables, and subsequently solving systems of linear equations arising from these payoff tables by various means [46][63]. These methods have relatively low A to I as a significant amount of time is invested by the researchers gathering and encoding problem specific expert knowledge to both define and solve the problem. It is likely that many real world problems are intractable by game theoretic methods. For the world leading Poker program mentioned in [63] developed at the University of Alberta GAMES group, the considerable amount of work that went into abstracting the game of Texas hold’em poker, effectively partitioning the partially observable game states into a game tree small enough to solve is apparent from [65], supported by personal discussions with members of the University of Alberta GAMES group. Game theoretic methods are typically parallelizable and scale well [63][66]. However, the results of learning are often numerical therefore difficult for humans to interpret.

Artificial Neural Networks (ANN) attempt to fit a modeling structure to a distribution of observations [43]. ANN offer flexible modes of training and operation in online, offline, supervised, unsupervised and reinforcement learning roles. Theoretically, ANN exhibit a high A to I ratio, however it is common engineering knowledge that a certain amount of expert tuning and experimentation is required for good results. ANN have had numerous successes in agent control problems [52][54][55] . However the result of learning is a table containing a series of weights which makes human interpretation of the results difficult. The parallelization of ANN for distributed learning is in its relative infancy, but shows some promising results [67].

Imitation learning [56] is more of a general approach to learning (a learner agent from a teacher agent) than a specific class of algorithm. It incorporates online and offline elements, as well as supervised, unsupervised, and reinforcement learning aspects into an overall approach. The model of learning is unique in that the model of learning is a robot or autonomous agent which imitates a teacher. Imitation is in its relative infancy compared to the more established approaches discussed above. This is evidenced by the widely different implementations utilizing different underlying methods, typically consisting of one or more expertly combined specific ML approaches [48][54].

Custom, one-off, or problem-specific approaches take many possible forms. They share one thing in common, however: a low A to I ratio because they are a form of parameterized expert system. Hence, we may refer to them as custom/expert approaches. Often these perform quite well since human problem solving ability still exceeds that of an artificial method for many non-trivial problems. In DAI applications this usually involves direct application of control (or other applicable) theory with hand tuned

parameters and control logic [57][58]. As formalized approaches (RL, evolutionary, game theoretic, etc.) increasingly make use of heuristics specific to the problem at hand, they begin to resemble custom/expert approaches. The extent to which the examples cited above actually belong in the custom/expert category is beyond the scope of this thesis, although it can be noted that [66] appears to lament the significant amount of human insight required to solve checkers.

2.5.3 Choosing a Learning Approach

The Automated Policy Optimization (APO) learning task discussed in sections 7.2, 7.3, and 7.5 involves optimizing a policy executed within an ABM of a hospital emergency department (ED). For the APO task, a reinforcement learning approach or style of learning is natural since there will be some performance feedback from the ABM (ruling out unsupervised learning), however the correct action to take in each state of the environment is not known (ruling out supervised learning). The performance of an agent executing a policy in this context is determined by the action taken by the agent given some state of the environment (the simulated ED) which closely resembles the classical RL problem formulation [59]. In addition to learning methods specifically tailored to RL problems, most notably Q-Learning, TD, or SARSA [59] the most popular types of learning for this class of problem found in the literature are biologically inspired – evolutionary learning [45][46][47], and artificial neural networks (ANN)[68] .

A popular area of interest in DAI has been concerned with robotics, specifically robot soccer [22], so many of the applications seen in the literature are drawn from the robot soccer domain. Robot soccer has much in common with APO task in that many robot

soccer applications involve generating policy for spatially oriented, embodied agents. Researchers involved in robot soccer often favor reinforcement learning specific methods [50] or evolutionary approaches [53]. Applications of ANN are also well represented [55]. Imitation learning [56] is an emerging topic of interest, however, questions remain as it is in its relative infancy compared to the rest of these approaches.

Although imitation learning [56] is likely amenable to RL tasks, some challenges are evident. Sharing of cultural information (such as by imitation) has the potential to improve performance of other ML approaches when they are used in concert [48][54][69]. However, it is unclear whether imitation on its own can improve upon the performance of the teacher on its own. To use imitation alone, such a teacher would have to be generated using human domain expertise input into the system. In effect the human would be manually training the learner. This is perhaps acceptable for simple motor tasks but not for complex optimization tasks such as the APO task defined in chapter 7.

Ultimately, an evolutionary approach, GP, was chosen because it has proven to be a successful approach in a variety of real engineering problems. Many such problems in [44] involve circuit design. The flexibility of GP to interface with already made tools such as circuit simulation tools allow for reuse of existing tools which is desirable from a Software Engineering perspective. GP is amongst the most flexible ML approaches as evidenced by numerous and diverse real-world applications [44][70].

In the context of A to I ratio, GP wins out over other applicable methods including GA since the alternatives would require the researcher to parameterize the policy in turn requiring considerable human analysis of the problem. In the case of the surveyed RL-specific methods the researcher must divide up the state space somewhat intelligently

when the state space is continuous or has a large range of integral values. As mentioned, in practice ANN require considerable human intuition, heuristics, and experience to tune, potentially impacting the expected high A to I of this approach. In comparison, when solving a problem with GP a practitioner needs to follow five "preparative steps", two of which are of a more administrative nature requiring little domain specific knowledge. For a more detailed discussion on why these steps are considered, to impart a minimal amount of human expertise, the reader is urged to consult [44].

1. & 2. Create a set of primitive functions (1) and terminals (2) which form the evolved program. This step imparts the most human domain expertise, and is a requirement of ML approaches in general. However, any flexible (i.e. re-usable, extendible, well engineered) ML software, including GP, can be interfaced with existing domain specific tools such as Integrated Circuit simulators, both promoting the reuse of these tools and the rapid development of the primitive functions and terminals.

3. Specifying the fitness measure is equivalent to specifying the problem to be solved, a step that is common to all learning algorithms which utilize a feedback signal.

4. Choose parameters for the GP run is administrative in the sense that in cases where the run time of the algorithm is dominated by fitness evaluation (as is the case with the tasks in Chapter 7) the parameters are a function of available compute time, wall time constraints, and execution time of fitness evaluations. This is contrasted with problem-specific tuning often required by approaches such as ANN.

5. Choose termination criteria. This step is closely tied to the fitness measure described in step 3, and is also administrative in nature.

As noted in chapter 7, the APO task is also "embarrassingly parallel" or easily distributable, therefore a good matching ML approach should be amenable to parallel execution. The benefits and ease of parallelizing GP are well documented and explored [44][71], while distributed learning in ANN and RL-specific algorithms are in relative infancy, but show some promise [60][67]. GP is the clear winner in terms of proven distributed scalability.

As discussed in Chapter 7, humans will have to interpret the results of policy optimization as part of a larger decision making process. Therefore the symbolic nature of GP is an asset which makes human interpretation of resulting policies more convenient when compared to others such as ANN, RL, or even GA which are a set of weights, essentially.

GP is not without its weaknesses, however. The first is that it is computationally intensive, or that learning takes a long time, a tradeoff for its generality. Another drawback from a theoretical perspective is the lack of formal proof for convergence, rather appealing to a more plausibly reasoned argument for its validation, spanning a corpus of successful applications [44][70].

2.6 References

[1] A. Uhrmacher and D. Weyns, Eds., *Multi-Agent Systems: Simulation and Applications*, New York, New York, CRC Press, 2009.

[2] J. Anderson. A generic distributed simulation system for intelligent agent design and evaluation. In *Proceedings of AI, Simulation and Planning In High Autonomy systems*, Tuscon, 2000.

[3] H.V.D. Parunak. "Go to the Ant": Engineering principles from natural multi-agent systems. *Annals of Operations Research*, 75(0):69-101, January 1997.

[4] B.P. Zeigler, T.G. Kim, H. Praehofer. *Theory of Modeling and Simulation.* Academic Press, Inc, Salt Lake, UT, 2nd edition, 2000.

[5] David Midgley, Robert Marks, Dinesh Kunchamwar. "Building and assurance of agent-based models: An example and challenge to the field". *Journal of Business Research*, Volume 60, Issue 8, Complexities in Markets Special Issue, August 2007, Pages 884-893, ISSN 0148-2963, DOI: 10.1016/j.jbusres.2007.02.004

[6] Windrum, Paul, Fagiolo, Giorgio and Moneta, Alessio (2007). 'Empirical Validation of Agent-Based Models: Alternatives and Prospects'. Journal of Artificial Societies and Social Simulation 10(2)8. [Online] Available: http://jasss.soc.surrey.ac.uk/10/2/8.html

[7] Robert Axtell, Robert Axelrod, Joshua M. Epstein, and Michael D. Cohen, "Aligning Simulation Models: A Case Study and Results". *Computational and Mathematical Organization Theory*, Vol. 1, Number 1, pp. 123-141, 1996.

[8] (Online). http://www.lanl.gov/programs/nisac/EpiSimS.shtml (Accessed: September 2010).

[9] Phillip Stroud & Sara Del Valle & Stephen Sydoriak & Jane Riese & Susan Mniszewski. "Spatial Dynamics of Pandemic Influenza in a Massive Artificial Society," *Journal of Artificial Societies and Social Simulation*, vol. 10. 2007. [Online] Available: http://jasss.soc.surrey.ac.uk/10/4/9.html

[10] G. Polya, *Mathematics and Plausible Reasoning: Volume 1 Induction and Analogy in Mathematics*. Princeton University Press, 1990.

[11] Carley, Kathleen & Altman, Neal & Kaminsky, B & Nave, D & Yahja, Alex. BioWar: A City-Scale Multi-Agent Network Model of Weaponized Biological Attacks. *Carnegie Mellon University, School of Computer Science, Institute for Software Research International, Technical Report CMU-ISRI-04-101*. 2004.

[12] Bonabeau, E., "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences*, 99 (3): pp. 7280-7287, May 2002.

[13] (Online). http://www.swarm.org/ (Accessed: September 2010).

[14] Volker Grimm, Uta Berger, Donald L. DeAngelis, J. Gary Polhill, Jarl Giske, Steven F. Railsback, The ODD protocol: A review and first update, *Ecological Modelling*, Volume 221, Issue 23, 24 November 2010, Pages 2760-2768, ISSN 0304-3800, DOI: 10.1016/j.ecolmodel.2010.08.019.

[15] T. Lethbridge and R. Lagraniere, *Object-Oriented Software Engineering: Practical Software Development using UML and Java*, McGraw Hill, 2001.

[16] J. Miller. "Active nonlinear tests (ANTs) of complex simulation models." *Manage Sci* 1998; 44(6):820-30.

[17] T. Schelling. "Dynamic Models of Segregation." *Journal of Mathematical Sociology* 1:143-186. 1971.

[18] R. Axelrod. THE EVOLUTION OF COOPERATION. Basic Books, 1984.

[19] Grimm and Volker, *Individual-based modeling and ecology*, Princeton: Princeton University Press, 2005.

[20] M. Borkowski, B.W. Podaima and R.D. McLeod, "Epidemic modeling with discrete space scheduled walkers: Possible extensions to HIV/AIDS," BMC Public Health, vol. 9 (Suppl 1): S14, 2009. [Online]. Available: doi:10.1186/1471-2458-9-S1-S14

[21] R.P. Hämäläinen and J. Parantainen. "Load analysis by agent-based simulation of the electricity distribution system". *Proc. of the 2nd IFAC Symposium on Control of Power Plants and Power Systems*, SIPOWER'95, Cancun, Mexico, December 6-8, 1995, Vol.I, pp. 213-217.

[22] (Online). http://www.robocup.org/robocup-soccer/simulation/ (Accessed: September 2010).

[23] (Online). http://sugarscape.sourceforge.net (Accessed: September 2010).

[24] "MPI Documents" (Online). Available: http://www.mpi-forum.org/docs/ (Accessed: May, 2010).

[25] R. Connell, P. Dawson, A. Skvortsov. *Comparison of an Agent-based Model of Disease Propagation with the Generalized SIR Epidemic Model*. Unclassified Technical Report DSTO-TR-2342, Defense Science and Technology Organization, Department of Defense, Commonwealth of Australia, 2009.

[26] Christiansen, J. H.; and Campbell, A. P, "HealthSim: An Agent-Based Model for Simulating Health Care Delivery," Argonne National Laboratory, Chicago, Illinois, 2003.

[27] US Defence Modeling and Simulation Office. HLA Interface Specification, version 1.3, 1998.

[28] F. Menczer and R.K. Belew. *Latent energy environments*. Technical Report CS93-301, University of California, San Diego, La Jolla, CA, 1993.

[29] (Online). http://wwwmosi.informatik.uni-rostock.de/mosi/projects/cosa/james-ii/ (Accessed: September 2010).

[30] (Online). http://www.simsesam.de/ (Accessed: September 2010).

[31] B. Roche, J.-F. Guégan, and F. Bousquet, "Multi-agent systems in epidemiology: a first step for computational biology in the study of vector-borne disease transmission", *BMC Bioinformatics*, vol. 9,p.435, 2008.

[32] (Online). http://repast.sourceforge.net/ (Accessed: September 2010).

[33] (Online). http://www.cs.bham.ac.uk/research/projects/poplog/packages/simagent.html (Accessed: September 2010).

[34] (Online). http://rtg.cis.upenn.edu/mobies/charon/implementation.html (Accessed: September 2010).

[35] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. "MASON: A Multi-Agent Simulation Environment." *Simulation: Transactions of the society for Modeling and Simulation International*. 82(7):517-527. 2005.

[36] (Online). http://ccl.northwestern.edu/netlogo/ (Accessed: September 2010).

[37] (Online). http://education.mit.edu/starlogo/ (Accessed: September 2010).

[38] (Online). http://www.spiderland.org/ (Accessed: September 2010).

[39] (Online). http://spades-sim.sourceforge.net/ (Accessed: September 2010).

[40] (Online). http://www.xjtek.com/ (Accessed: September 2010).

[41] David Harel, Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, June 1987.

[42] (Online). http://setiathome.berkeley.edu/ (Accessed: September 2010).

[43] T. Mitchell, *Machine Learning*. Boston, MA: McGraw Hill, 1997.

[44]] J.R. Koza, et al., *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. New York, NY: Springer, 2005.

[45] N. Vlassis, A concise Introduction to Multiagent Systems and Distributed Artificial Intelligence, Part of Synthesis Lectures on Artificial Intelligence and Machine Learning, R. Brachman and T. Diettrich, Ed. Morgan & Claypool, 2007.

[46] Y. Shoham, K. Leyton-Brown, Multiagent Systems: Algorithmic, Game Theoretic, and Logical Foundations, Cambridge: 2009.

[47] S. Calderoni and P. Marcenac, "Genetic Progamming for Automatic Design of Self-Adaptive Robots," *EuroGP 1998*: pp. 163-177. 1998.

[48] J. Noble and D. Franks, "Social Learning in a Multi-Agent System," *Computing and Informatics*, vol. 22, 2004.

[49] M. Mataric. "Reinforcement learning in the multi-robot domain," *Autonomous Robots*, 4(1):73–83, 1997.

[50] M. Littman, "Markov games as a framework for multi-agent reinforcement learning," In *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 157–163, San Mateo: Morgan Kaufman, 1994.

[51] P. Nordin and W. Banzhaf, "An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming," *Adaptive Behavior*, vol. 5, pp. 107-140.

[52] J. Liu, Autonomous Agents and Multi-Agent Systems: Explorations in Learning, Self-Organization and Adaptive Computation, Singapore: World Scientific, Singapore, 2001.

[53] D. Andre and A. Teller, "Evolving team Darwin United," In *RoboCup-98: Robot Soccer World Cup II*, M. Asada and H. Kitano, eds., Berlin: Springer Verlag, 1999.

[54] A. Acerbi and S. Nolfi, "Social Learning and Cultural Evolution in Embodied and Situated Agents," *Proceedings of the First IEEE Symposium on Artificial Life*, New Jersey: IEEE Press, 2007.

[55] P. Stone and M. Veloso, "Towards collaborative and adversarial learning: A case study in robotic soccer," *International Journal of Human-Computer Studies*, 48(1):83–104, January 1998.

[56] M. Mataric. "Sensory-motor primitives as a basis for imitation: linking perception to action and biology to robotics," *Imitation in Animals and Artifacts*, K. Dautenhahn and C. Nehaniv, eds., pp. 391–422. MIT Press, 2002.

[57] Santos J. M., Scolnik H. D., Laplagne I., Daicz S., Scarpettini F., Fassi H., Castelo C. "UBA-Sot: An approach to Control and Team Strategy in Robot Soccer", *International Journal of Control, Automation, and Systems*, 1:1, pp. 149-155. 2003.

[58] M. Sahota, "Reactive deliberation: An architecture for real-time intelligent control in dynamic environments," In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 1303– 1308, 1994.

[59] R. Sutton, A. Barto, *Reinforcement Learning: An introduction*. Cambridge: MIT Press, 1998.

[60] Kushida, M., Takahashi, K., Ueda, H., and Miyahara, T. A Comparative Study of Parallel Reinforcement Learning Methods with a PC Cluster System. In *Proceedings of the IEEE/WIC/ACM international Conference on intelligent Agent Technology*

(December 18 - 22, 2006). IAT. IEEE Computer Society, Washington, DC, 416-419. 2006.

[61] V. Babovic and M. Keijzer, "Genetic programming as a model induction engine", *Journal of Hydroinformatics*, vol. 2, no. 1, pp. 35-60, 2000.

[62] M. Brameier, W. Banzhaf, *Linear Genetic Programming*, New York: Springer, 2007.

[63] M. Zinkevich, M. Johanson, M. Bowling, C. Piccione, "Regret Minimization in Games with Incomplete Information", In *Advances in Neural Information Processing Systems 20: 21st Annual Conference on Neural Information Processing Systems*, 2007

[64] (Online). http://www.research.ibm.com/deepblue/ Accessed December 2010.

[65] D. Billings, "Algorithms and Assessment in Computer Poker," Ph.D. dissertation, University of Alberta, Edmonton, AB, Canada, 2006.

[66] J. Schaeffer, *One Jump Ahead: Challenging Human Supremacy in Checkers*, Springer, 1997.

[67] Calvert, D. and Guan, J. 2005. Distributed Artificial Neural Network Architectures. In *Proceedings of the 19th international Symposium on High Performance Computing Systems and Applications* (May 15 - 18, 2005). HPCS. IEEE Computer Society, Washington, DC, 2-10.

[68] I. Ulusoy, M. Parnianpour, N. Berme, S. Simon, "A neural network system with reinforcement learning to control a dynamic arm model," *Biomedical Engineering-Applications, Basis & Communications*, vol 13, no. 3, 2001.

[69] L. Spector and S. Luke. "Culture enhances the evolvability of cognition". *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*. Mahwah, NJ, 1996.

[70] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[71] W. Bhanzhaf, P. Nordin, R. Keller, F. Francone, *Genetic Programming – An Introduction: On the automatic evolution of computer programs and its applications*. Morgan Kaufmann Publishers, Inc., 1998.

Chapter 3

Development of an Agent Based Simulator for Systems Modeling

This chapter describes the first version of the Agent Based Model (simulator) for modeling the Hospital Emergency Department (ED) system described in Chapter 1. This represents an initial attempt towards developing the ABM software framework, as well as the accompanying conceptual framework. Through this process intuition is gained towards the decision support tool proposed in Chapter 1. This Agent Based Model (ABM) is a model of an ED and the agents represent the doctors, patients and nurses. The particular ABM in development is closely integrated with a visualization component, that as will be seen in the next chapter enhances the ABM as a communication tool between the modeller and any healthcare practitioner or administrators involved.

The ABM in the stage of development described here is a proof of concept that an ED can be modeled effectively in such a way. Additionally, it sets the stage for future extensions which develop the model into a useful tool for modeling "what-if" scenarios, and eventual integration with a Machine Learning module for automated policy generation. In the former case, administrators might use such a tool to test a proposed

policy without implementing it and incurring the expense and potential uncertain impacts on patient care. As part of a plausibly reasoned [1] argument, improvement in the simulated ED in response to a policy change adds credibility that the policy is a reasonable one to try in the actual ED. In the latter case, administrators could pro-actively use an ABM tool augmented with Machine Learning technology to search for policy improvements within a set of constraints.

The following is an augmented version of "Agent-based simulation of emergency departments with patient diversion" by S. Mukhi and M. Laskowski [2]. This author's contribution was the design and implementation of the ABM discussed, running the simulation, analyzing the results, as well as the written portions of the above paper specific to the ABM. In addition, the policies for patient diversion and computer experiments were developed and carried out by the author.

3.1 Agent-based Simulation of an ED with Patient Diversion

An Agent Based Model of an emergency department and its utility for evaluating workflow and assessing patient diversion policies are described in this chapter. The overall goal of the research is to develop tools to better understand and manage emergency departments. There are several modes in which Agent Based Modeling tools may be of benefit. In a self contained manner the operation of an emergency department can be modeled. In this mode, policies such as staffing could be changed and the effect

on parameters such as waiting times and throughput could be quantified. In an extended version, multiple emergency departments can be modeled and would allow for the evaluation of ambulance or patient redirection policies. In either case it is reasonable to suggest an effective means of augmenting the simulation with empirical data collected using a proximity location and tracking system within an emergency department. This Agent Based Model allows for a simulation of a number of emergency departments and introduces a method of extracting real time patient data from emergency departments throughout a city allowing for the evaluation of patient diversion policies.

3.1.1 Introduction

Hospitals represent one of the most promising areas where Agent Based Modeling may be seen as an effective tool in evaluating policy and improving efficiencies. In many cases, the operations of an emergency department are over taxed and may not be guided by optimal policies. Although policies evolve over time and efforts are made to reduce wait times etc. often there is little quantitative analysis or feedback in the process. Using Agent Based Modeling an emergency department can be modeled and in this manner better use of resources can be made possible through identification of anomalies and bottlenecks which may be difficult to detect otherwise.

In addition, technologies are emerging that will be leveraged by Hospitals to improve patient care. Two of the more obvious technologies and applications include inter Hospital tracking and internetworking. These technologies will also allow for a more distributed approach to managing a number of interacting emergency departments. Along

these lines the emergency department ABM can be used to evaluate ambulance redirection or diversion policies.

The research presented here provides a specific emergency room example application and an extension to a wide area Hospital/ED/Ambulance and patient diversion scenario. The remainder of this chapter discusses work completed to date on an open source visualization, simulation, and wait time forecasting simulation suite; then presents simulation results for the scenarios discussed above, for use in evaluating workflow within an ED, and for use in patient diversion policies.

3.1.2 Applications

The applications discussed below are of increasing interest here and elsewhere. Although Agent Based Modeling of an emergency department is of utility on its own [3], it is of considerably more utility when combined with tracking technologies and networking capabilities. Specifically, the Agent Based Model is a distributed model across a number of regional hospitals with emphasis on utilizing data collected in real time. Another novelty is the use of congestion avoidance algorithms from Telecommunications Engineering redeployed as a model for evaluating patient diversion policies. The following applications are included as context for which the emergency department ABM is well suited.

3.1.3 Local Area Patient Tracking

An application being developed is directed towards monitoring or estimating the location of patients as they enter an emergency room until treated and released. This is an extremely active area of research and concern for both the general public as well as practitioners. In the following scenario, emergency department simulation is enhanced with empirical data collection technologies [4].

In one scenario a patient is tagged with an RFID tag when registering at the emergency department, and an RFID reader at the desk records the event. As the patient enters the waiting area another reader reads this event. The event and reader location are recorded at an Hhost (Hospital host subsystem). Data logged by the Hhost (timestamp, tag and reader ID) would be uploaded via the nearest Wi-Fi access point to be relayed to CORE (Central Observation and Reporting Environment). Once stored at the CORE it can be made available for augmenting the ABM, mining or made available over the Internet to other stakeholders. In certain circumstances the information can be made available to the public such that they may be better informed as to which emergency department or clinic they may choose to attend. A specific example would be information concerning wait times at facilities for injuries such as broken bones and whether or not a physician was on staff to set a break at that time.

Figure 3.1 illustrates an emergency department with RFID readers deployed strategically throughout the area capable of reading and logging patient tags for subsequent uploading to a more central network service.

In the future it is also conceivable that a patient's tag could be updated during treatment such that a more complete record of a where bottlenecks occur could be extracted. For example, upon receiving an X-ray the RFID tag could be programmed to store this information and subsequently uploaded to CORE when read by the next reader encountered. The tag could also be programmed at admitting indicating the general type of aliment or complaint for more complete although anonymous data collection and analysis.

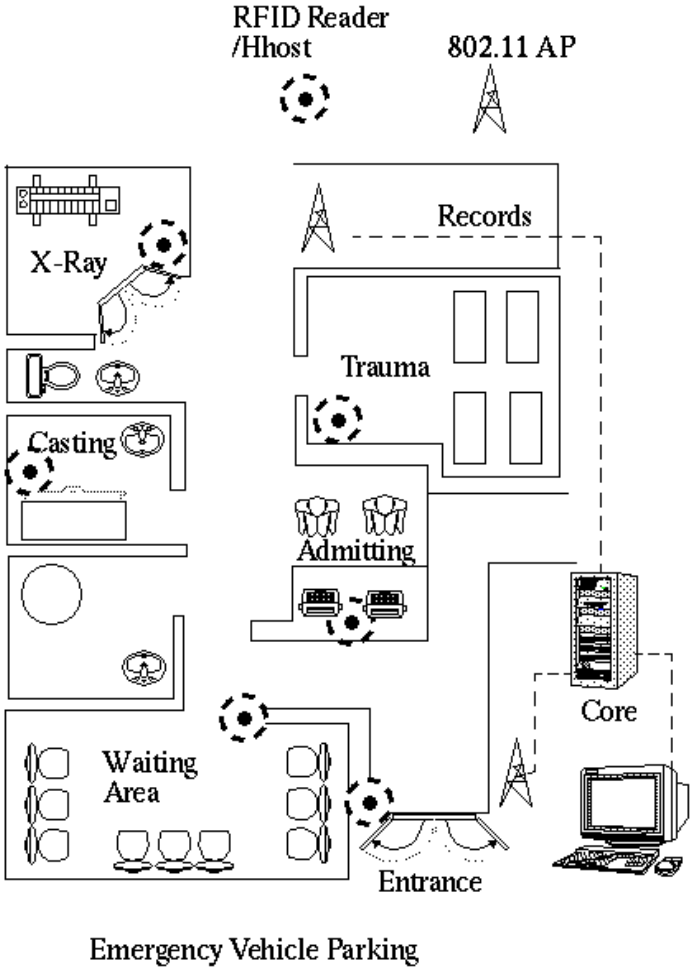

Figure 3.1. Schematic of tagging locations in a Hospital Emergency Facility

3.1.4 Wide Area ED Scenario

In a wide area hospital/ED/ambulance scenario the basic CORE functionality would scale and encompass alternative communication modalities. At present, this prototype CORE is designed to support wide area networking having evolved from a geographically dispersed event management system [5]. Also as CORE is IP centric, security issues are largely resolved using VPNs and encryption standards associated with IP. The following figure illustrates the basic components and agents in the extended framework. Figure 3.2 illustrates a wide areas scenario incorporating participation hospitals and emergency service vehicles.

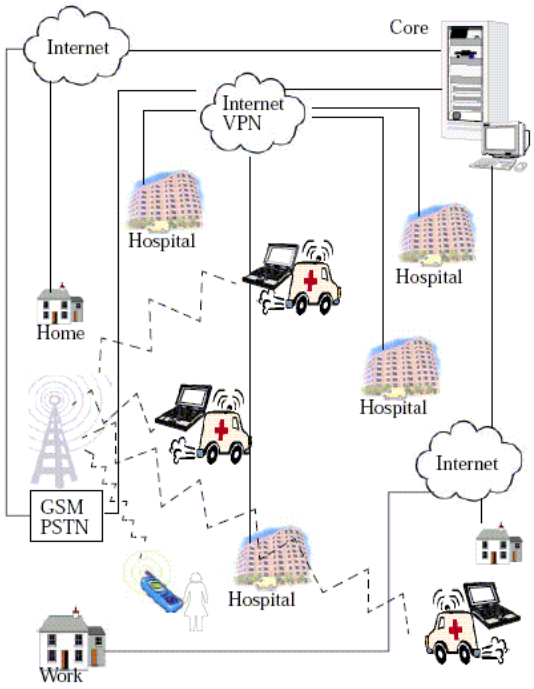

Figure 3.2. Wide area deployment of the framework, illustrating the major stakeholders or agents.

In the wide area scenario (Figure 3.2) each hospital emergency department would be equipped as in Figure 3.1 with data extracted from RFID proximity location systems augmenting the Agent Based Modeling.

More proactive modeling extensions include the ability to notify and receive information from ambulances and other emergency vehicles. The actual communication services likely would be over GSM or similar communication infrastructures. Here these services can be modeled as messages between agents and the platform would be used to assist in optimizing ambulance diversion policies. Other types of considerations required would be in the estimation of travel time as these factors would significant in an effective model. Although not addressed here, with the proliferation of GPS and mapping technologies, these estimates can again become empirical inputs to the multiple emergency department simulation.

At present ambulance diversion is principally based on best effort reporting and operating in good faith based on regional guidelines, an example of which can be found at [6]. It is suggested that in addition to these heuristics, emergency department modeling can benefit from algorithms more commonly associated with the Internet and congestion avoidance schemes that deal with overcrowding of routers. As an example, the Random Early Detection (RED) algorithm [7] is adapted as a candidate for consideration when attempting to optimize ambulance or patient redirection. This is an ideal initial algorithm for adaptation as it has many of the attributes well suited to improving system throughput. RED accommodates limited bursts and can be effective even when there is limited sharing of information between emergency departments. In this case, RED would

be used as a model for redirection based on emergency department congestion information being made to potential patients/ambulances.

3.1.5 Visual Simulation Suite

As mentioned in section 3.1.2, the ABM is an object oriented (OO), open-source, visual simulator being developed which could be used with data gathered from the previously discussed architecture, and may be used to analyze and forecast emergency department waiting times. The simulator was written using C++ and makes use of the Qt4 API for cross platform windowed applications [8]. By virtue of Qt4 itself being open-source as well as the simulator code, once the source is released in Beta stage, this project will benefit from other researchers customizing and extending its code. Neither of these things would be possible had an off-the-shelf proprietary solution been used instead of an open-source paradigm. Qt4 also enables deployment of the simulator on Windows, Mac, or Linux. A screenshot of the simulator window is shown in Figure 3.3. As will be demonstrated in later chapters, more detailed and sophisticated visualization modules have been developed since this initial proof of concept. The simulated world is a two-dimensional (X,Y) discrete Cartesian world of extremely high resolution (floating point). The spatial aspect of the visualization reflects the spatial nature of the underlying data that will be collected. The spatial aspect of the model will be important in Chapter 6 when the framework is extended to model the spread of an Influenza Like Illness causing virus within the emergency department.

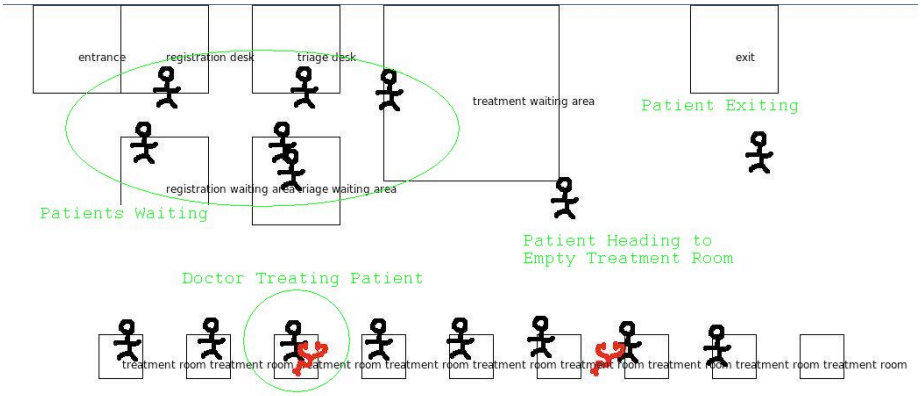

Figure 3.3. Screen capture of the basic simulator.

3.1.6 Emergency Department Model

The aspects of the emergency department treatment process of interest to the model are depicted in Figure 3.4. Patients arrive either by ambulance or walk in. Patients in need of immediate care are sent straight through to the treatment area. All ambulance arrivals are considered to be in need of immediate care, as well as some small fraction of walk-in patients.

Walk-ins that do not require immediate care proceed to the registration desk. If the registration desk is busy with an earlier arrival, the arriving patient then waits in a queue. Once the registration process is complete, the patient proceeds to the triage station. Again, if the triage station is busy, the patient waits in a queue. The nurses at the triage station will assign the patient a priority based on the severity of their condition. The arriving patient then waits with other patients in what is effectively a priority queue to be assigned a treatment room.

Once assigned a treatment room the patient waits for a doctor to come around in order to receive treatment. It is assumed that doctors will treat the patients in order of urgency, and then in order of arrival. Upon completion of the treatment, the patient leaves the system, and both the treatment room and the doctor become available for another patient.

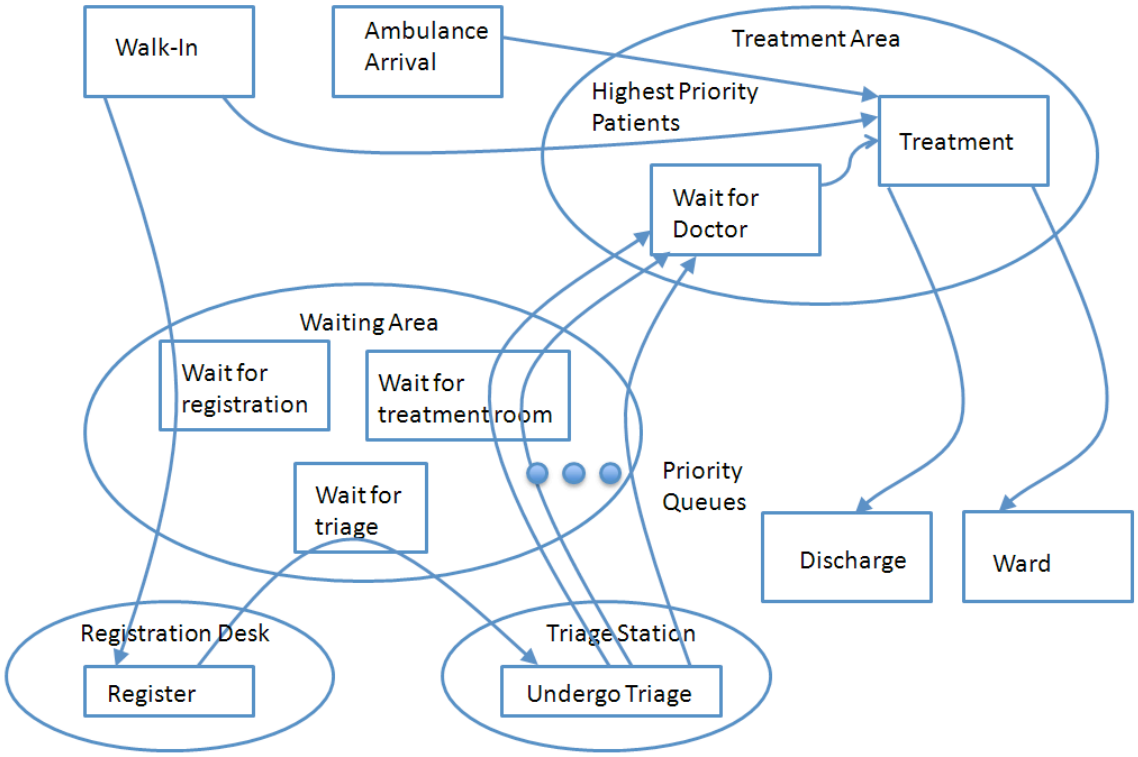

Figure 3.4. Model of Emergency Department Patient Service

3.1.7 Basic Architecture

The simulator maintains an instance of the erWorld class which keeps references to all the simulated objects described in this section. Each erWorld represents one emergency department. To implement the Wide Area Scenario to simulate and investigate ambulance diversion policies suggested in section 3.1.2, and further discussed in section

3.1.4, numerous instances of erWorld would be required. The flexibility and reuse of code made possible by the OO architecture makes this possible by subclassing or extending existing classes to allow communication between instances of erWorld. As mentioned, each erWorld maintains a collection of patient generators, agents representing nursing stations (registration, triage), patient agents, and doctor agents which represent corresponding aspects of the model discussed earlier. A special erController agent is used to mediate patient flow through the emergency department process. Creating subclasses of erController is necessary to be able to handle variations on the basic emergency department processes in order to reflect different policies for individually modeled emergency departments. For example, one would create a subclass of erController for an emergency department that allows for bedside registration for all patients versus an emergency department that requires most patients to register at a desk (as per current implementation). The classes that represent doctors and the nursing stations may also be subclassed to reflect procedures that vary between emergency departments. Patients too can be subclassed should the need arise.

As shown in Figure 3.3, functional areas of the emergency department can be placed in arbitrary locations. However, a worthwhile goal is to eventually overlay the locations on the actual floor plans of the simulated emergency departments. Referring to Figure 3.5, at every simulated time step of one second, all relevant agents are refreshed. In more detail, the main simulation loop calls the Advance() method in each simulated erWorld, the semantics of which are that time is advancing in that world by one second. Further hierarchical decompositions are possible, but regardless of the details, each agent has its time advanced in turn by having its tick() method called. Thus, one second at a time,

patients move between nursing, waiting, and treatment areas, as well as track the time spent in each activity. Doctors move to occupied treatment rooms and treat the patient within. Nursing stations count down the time required to process patients for the relevant activity. Patient generators model a Poisson arrival process for each patient class (i.e. classified by urgency of care required). At each time step each patient generator decides whether to introduce a new patient. At the proof of concept stage, data has yet to be collected, and thus arrival rates and service times are based on estimates obtained by other researchers [9]. However, the goal is to have the simulator driven by real world data, collected in real time, such as that proposed in section 3.1.2 where RFID proximity location system data would be used to augment the simulator. It is illustrative here to recall that the dynamics and complexity of the simulation are emergent from interactions between agents.

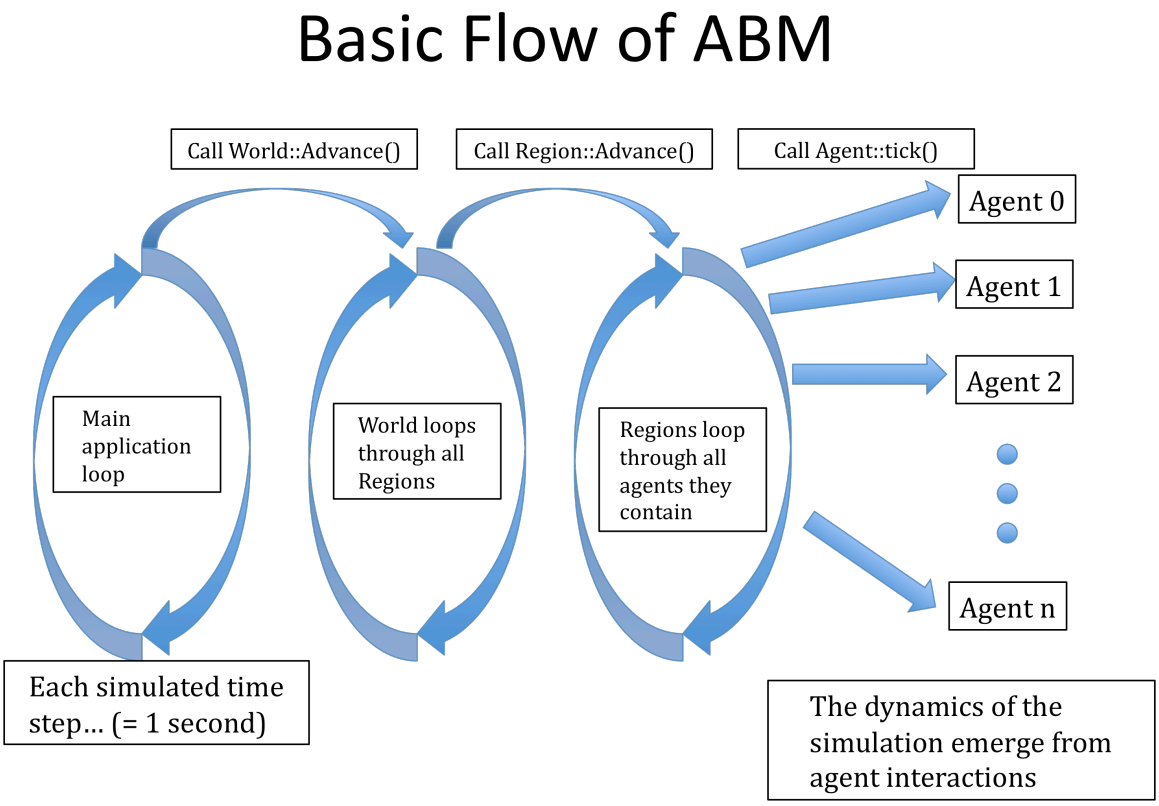

Figure 3.5. Simulator program flow

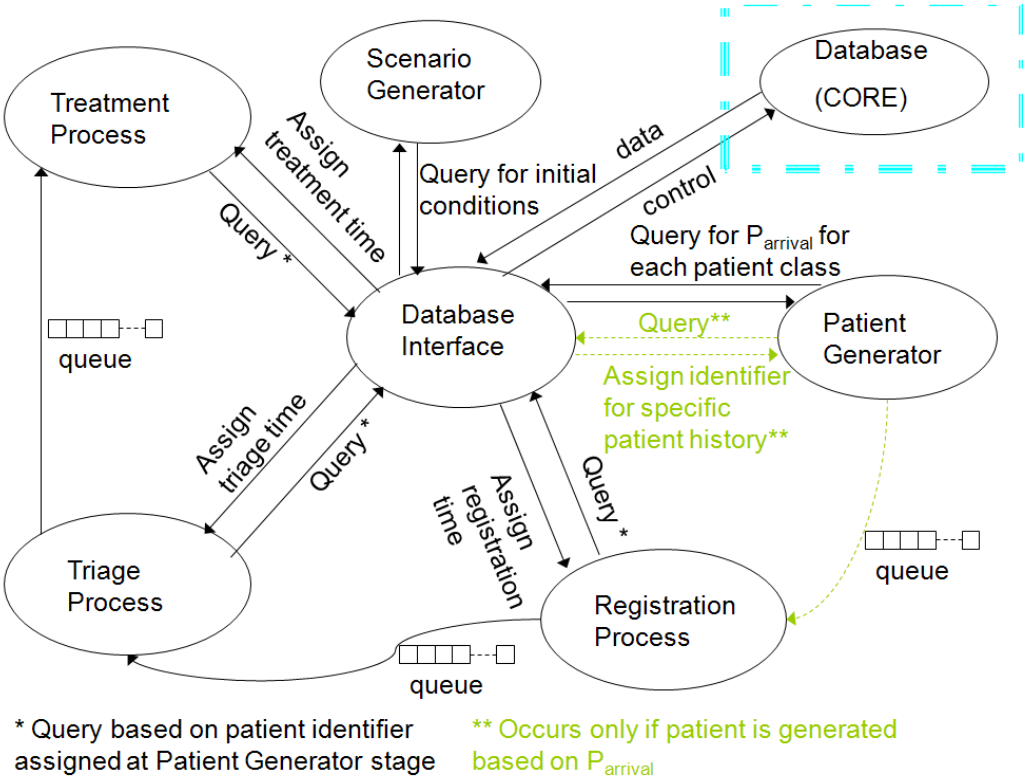

Figure 3.6. Data Flow for Simulation

3.1.8 Data Driven Simulation

As with all tools, there has to be a means of analyzing or incorporating real data. Currently deployed Emergency Department Information Systems (EDIS) suffer from inaccuracies mostly resulting from poor data recording habits of the staff doing data entry. Utilizing RFID technology would potentially allow for positional data for each patient and staff member in the system to be mined to infer the state of each.

Once collected, and after some additional processing, data can be used to drive the simulator by employing an intermediary database interface class, which insulates the

simulator code from the specific database implementation used. Processing may include sanitizing the data to ensure patient anonymity, and determining the arrival rate for the appropriate time of day and time of year. Figure 3.6 illustrates how the gathered data may be used during the course of the simulation.

When the simulation is started, the state of the simulated emergency department is seeded with the current state of the actual emergency department as inferred from the most recent data sent to CORE from the MHosts. The Patient Generator periodically queries for current arrival rates for each simulated patient class as illustrated in Figure 3.6. Those rates are used at subsequent time steps to determine whether a patient of a particular class arrives at that time step. If a patient arrives the agent is assigned a random patient history of the appropriate patient class upon creation and that history is used to determine service and treatment times for that patient. Note that times spent in queues are determined by the current state of the simulator, i.e. the number of patients waiting ahead of that patient, rather than the patient history.

In order to forecast future patient wait times, the simulation can be run into the future a number of times, keeping track of the wait times experienced by patients arriving at future times – until some reasonable level of confidence is reached. During this process the visualization can be disabled in order to speed multiple trials.

Prediction based on modeling and simulation is extremely difficult and potentially error prone. Confidence can be enhanced as the system is in operation and predictions tracked. As mentioned, a conjecture of this thesis is that the model of individual or interacting emergency departments augmented with whatever available empirical data is available would still be better than best intent open loop policy decision making.

3.1.9 Simulations

In the following sections two experimental results are presented that suggest the utility of the discussed ABM/simulator for making informed policy decisions. The first scenario investigated, while simple, illustrates the effects a policy decision has, such as changing staffing levels, using multiple performance metrics. In the second scenario, despite not having an actual RFID data collection system in place, an attempt is made to model what impact it would have if the infrastructure were in place to gather and disseminate ED utilization in real time. This would have the effect of informing the Ambulance Service as well as individual citizens (perhaps through a web portal) of the real-time status of EDs in the city in order to make better informed decisions about which ED to visit (based on current and projected wait-times).

3.1.10 Staffing Change Scenario

The basic ED scenario mentioned above is simulated, and with Triage Classes, Service Times, and Patient Arrival rates based on [9], comparing 3 different staffing scenarios of two, three, and four doctors working in the ED. The simulation was allowed a "warm-up" period of 24 hours, then observations were made during the following 24 hours. Ten independent trials were run, treatment queue length and doctor utilization were averaged, while individual patient waiting times are shown un-aggregated. These results are presented in figures 6, 7, and 8, respectively.

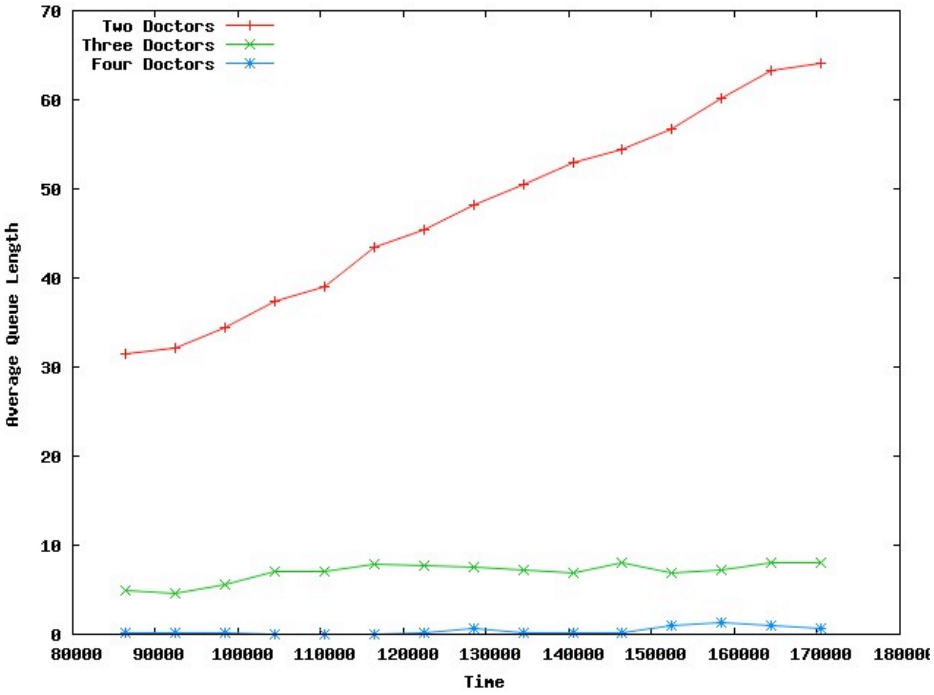

Figure 3.7. Average Queue Lengths for Varying Number of ED Doctors on Duty

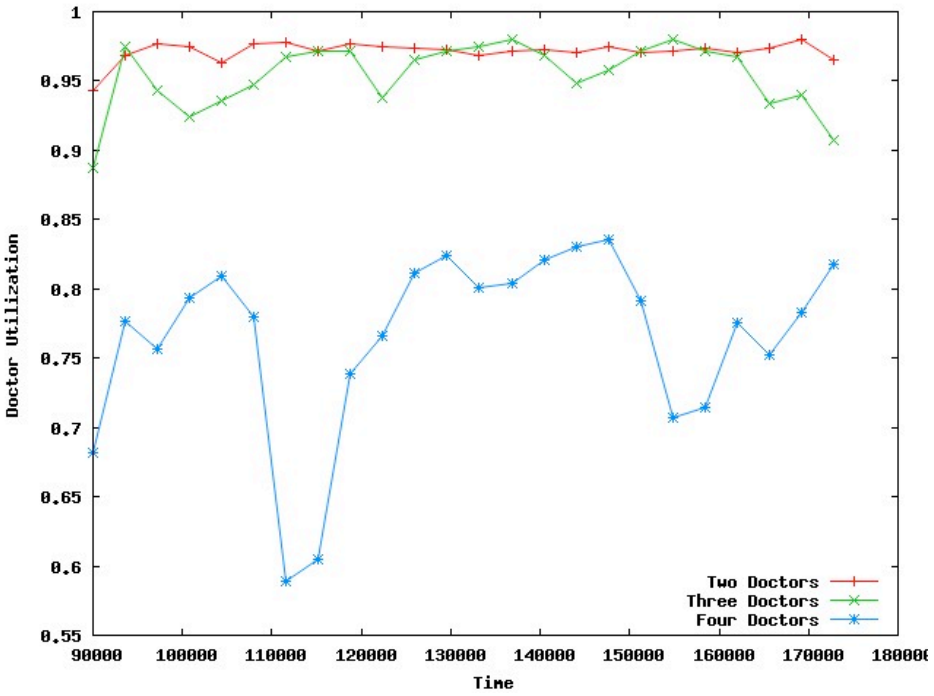

Figure 3.8. Average Doctor Utilization for Varying Number of ED Doctors on Duty

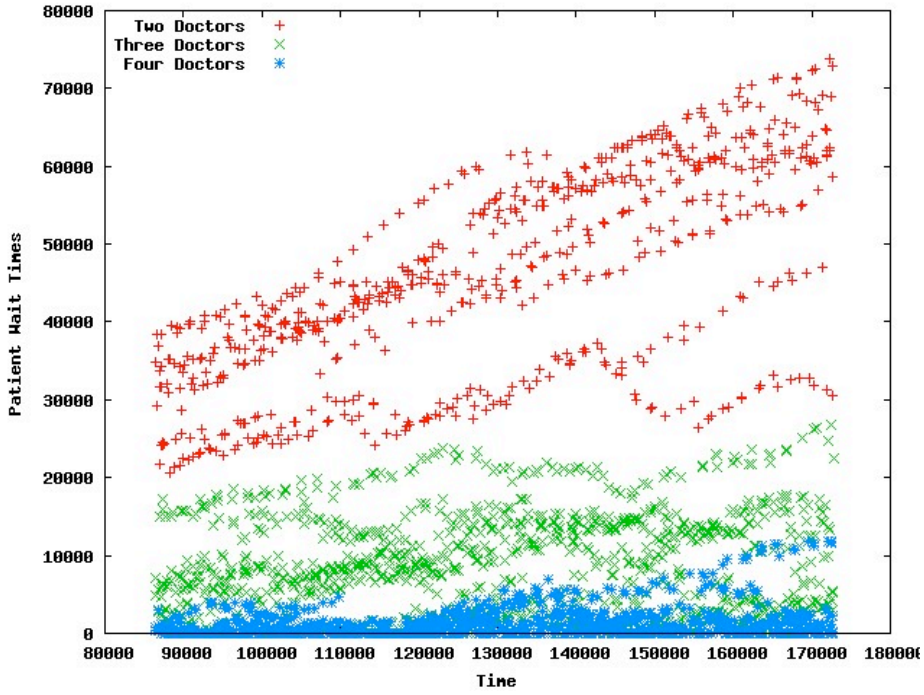

Figure 3.9. Patient Waiting Times for Varying Number of ED Doctors on Duty

From these figures one can see that intuitively the results seem reasonable. Figure 3.7 shows the average number of patients waiting for treatment as a function of time (in seconds). For example, as seen in the case of an ED with two doctors the patient queue continually increases with time as the ED is clearly under-staffed. In an ED staffed with four doctors the patient queue is nearly zero: however, Figure 3.8 illustrates that the doctors are underutilized. This suggests that for this scenario, unless it is critical to have nobody waiting for service, resources would be better allocated elsewhere rather than adding a fourth doctor. It is interesting to note that in Figure 3.9 each data point is an individual patient, albeit simulated, passing through the ED - in which case outliers would perhaps correspond to patients which waited an unusually long or short time. For

policy makers, it may be easier to empathize with individual patient cases than with a standard deviation or other such statistical measure.

3.1.11 Data Infrastructure Scenario

This section discusses modelling the impact of collecting real-time ED status using the RFID based framework discussed in section 3.1.2, then making available the real-time ED status to Ambulance operators and the public at large on a city-wide scale. Since this system is not yet in place locally, the decision was made to borrow from computer network engineering the well established technique of Random Early Detection (RED) [7] to model this process. Random Early Detection is a method of network congestion management, whereby senders of data over the network (typically the Internet) are implicitly notified of network congestion by having their data packets (data over the Internet is divided into discrete chunks, called packets) probabilistically dropped from network queues. To avoid oscillation between intense bursts of traffic and choking off traffic entirely, the rate at which these packets are dropped is ramped up gently after a certain threshold in the queue length is reached. Similarly, in this ED model, a minimum threshold is set, below which ED queue lengths are considered acceptable by everyone and no dropping occurs. The rate at which patients are dropped increases linearly with queue length until some maximum threshold is reached, past which the drop rate remains constant. Since "dropping", or turning away patients, is considered as unacceptable the model instead considers a drop to be a patient being redirected to another ED. The mechanism for this is either self-redirection to another ED or an Ambulance being redirected by a central dispatcher. Two modes are considered, one where patients are

redirected to a random ED with uniform probability, and one where patients are probabilistically redirected to an ED based on the ratio of doctors to patients waiting. In the latter case, this results in EDs that are less busy having a higher likelihood of patients being redirected there. This reflects an assumed patient preference for shorter waiting time, and also demonstrates the utility of having city-wide ED status information disseminated. This is contrasted to the former case, where patients are simply guessing as to which ED is a more preferable alternative without any guidance whatsoever.

To ground simulations as much as possible in reality, a report on Emergency Department usage in Winnipeg released just prior to this writing by the Manitoba Centre for Health Policy is cited [10]. There were 185,659 ED visits in Winnipeg among six Hospitals, the breakdown of which by CTAS [11] triage level roughly corresponds to the triage levels used in [9]. Since no data is available on treatment times it is plausible to use the distribution of treatment times based on triage level from [9]. At the time of this proof-of-concept, no data on variation in patient arrival rate, based on time of day, day of the week, time of year, or variation between individual EDs has been made available so the rates are assumed to be uniform for these variables. It should be noted that these types of variations can be easily incorporated in the simulator once data is available. With the information presented above, it was possible to estimate arrival rates of patients for each triage level at each simulated ED. An arbitrary but reasonable minimum threshold of 10 patients waiting in the queue was chosen for the RED model. The drop or redirection rate increases linearly to a maximum of 50% reached at a queue length of 20. It is not unreasonable to assume that staffing levels at each ED do not match demand. Because arrival rates are uniform among the simulated EDs, to make the simulation

interesting, two EDs are staffed with two doctors, two EDs have three doctors, and two EDs have four doctors on staff during the simulation.

As in the previous section, three 24 hour scenarios were investigated with ten trials each, and a warm up time of 24 hours. For comparison, the first scenario, called No Redirection, assumes that there is no ED status information available and that patients are better off going to the nearest ED and remaining there no matter what the wait. The second is the mode where redirection occurs based on the discussed RED model, and the destination ED is chosen from a uniform probability, referred to as Random RED. The third is the form of RED redirection where EDs with lower expected waiting time are probabilistically chosen more often as the destination ED. This case is referred to as Guided RED.

Un-aggregated patient wait times are not shown for these scenarios. The reason is because of the disparity between ED conditions, patient wait times vary wildly. However, Figure 3.10 illustrates that average queue lengths among all hospitals are shortest for Guided RED, which is an intuitively reasonable outcome and lends credibility to the modeling approach. Also, in Figure 3.11 overall doctor utilization is highest in the Guided RED scenario. The improved doctor utilization results in significantly reduced queue lengths or equivalently reduced patient waiting times. It is interesting to note that a significant queue length reduction (waiting time) was achieved with only a modest increase in utilization and no additional resources.

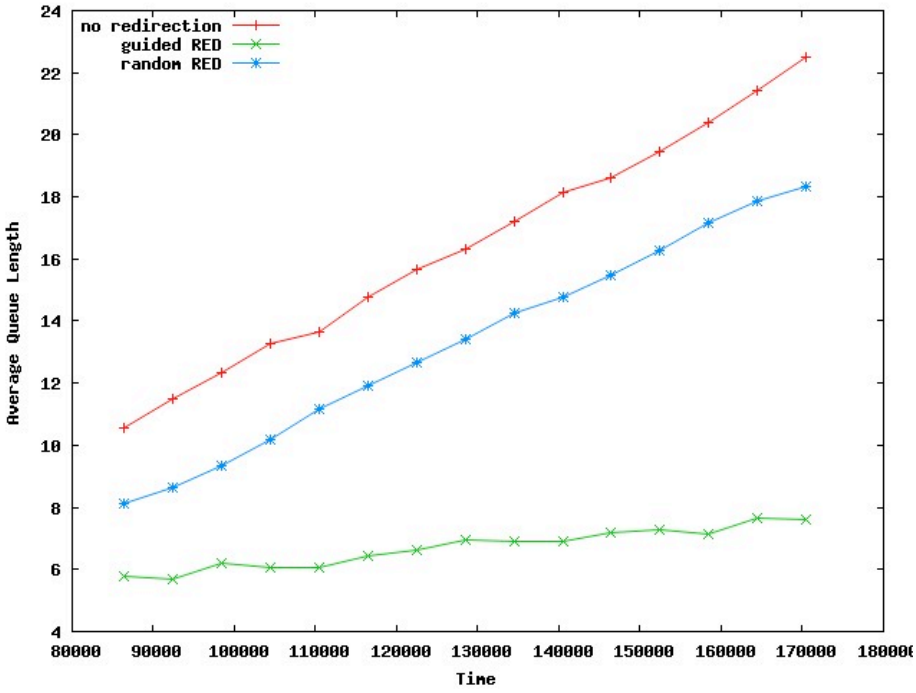

Figure 3.10. Average Queue Lengths for Various Redirection Policies

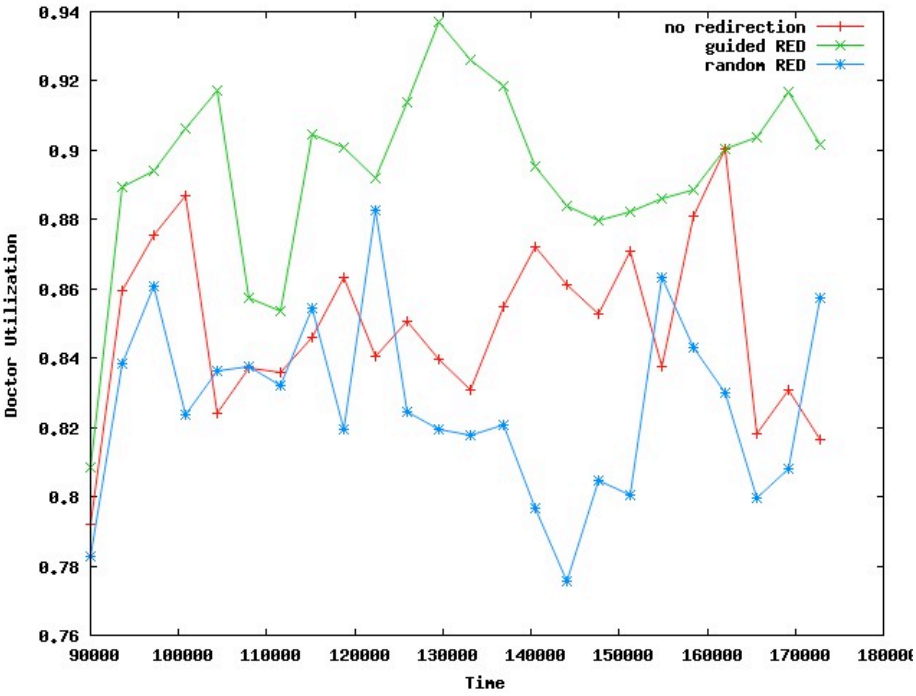

Figure 3.11. Average Doctor Utilization for Various Redirection Policies

3.2 Summary of Chapter 3

This chapter presented the initial development of an Agent Based Modeling system oriented to the simulation of emergency departments in either stand alone mode, multiple interacting emergency departments, as well as technologies well suited to enhance simulation with statistical empirical data collected in real time. Concepts from Telecommunications Engineering are introduced as a model for policy change regarding patient redirection. Simulation alone is not sufficient due to the difficulty in predicting highly uncertain patient arrival rates whereas the system presented here is oriented to augmenting simulation with empirical data when available. As such, the context of the work here also presented a means where emerging technologies such as RFID are suggested as data sources. These sources can be mined in a statistically significant manner and provide real world input for the simulation. The system being developed is also open source and relies on open source components. It is extendable and can be ported or tailored to a variety of hospital IT applications several of which were identified here.

Much of this chapter represents a first step towards the decision support tool proposed in Chapter 1. It is also an instance of an Agent Based Model used to model an emergency department as a system of agents governed by rules, with the complex dynamics of the system being emergent from the interactions between agents.

The early version of the ABM ED Simulator discussed here was focused on validating the concept of an ABM for healthcare applications, however a handful of

what-if scenarios were also investigated. It was not practical to set up varying scenarios at this stage of development because the simulator needed to be recompiled for each change. The scenarios investigated were different staffing levels, as well as a patient redirection policy between multiple ED's in a greater healthcare region. The discussion above serves as an illustration of how ABM fits into a policymaking decision process. The intuition gained by the research of this chapter was that ABMs appear to hold significant potential and have a significant role to play in healthcare applications. It was also evident that the problem is inherently complex and made more so by realizations of best practices in developing ABMs for healthcare. Some of these best practices included realization that there is a need to incorporate real data to the degree possible, to model to as high a degree possible the fidelity of the social dynamics and topographies of the ED, and the realization that there is a significant role for statistical processing.

In addition to the above intuition, opportunities for the application of Machine Learning within an ABM based decision support tool have been demonstrated – specifically, optimization of staffing levels, and automatic induction of a patient redirection policy. Owing to their social nature, such problems are "messy"; doctor utilization can be tied to a dollar cost, but this must be balanced against patient care which is difficult to quantify in a dollar amount. Furthermore, the modeled system modeled is stochastic in nature and unpredictable. This makes validation of the model, in the classic sense, difficult if not impossible. It is likely, although unproven, that modeling institutions such as EDs are computationally irreducible in the Wolfram sense [12]. This chapter represents one of the the first times an ABM was used in the context of a decision support tool for healthcare. This chapter also represents one of the first

examples of borrowing a well vetted telecommunication packet flow policy for use within a modeled healthcare patient redirection or diversion application.

The spatial aspect of the simulation is predicated by the expected integration with real time patient location system (RTLS) and improved electronic medical records systems. Since this work started, a moderate amount of progress has been made locally towards deploying such systems. There is some discussion concerning integrating this simulation with a data source. Additional details will be provided as to patient tracking technologies and data acquisition methodology in the next chapter. Although strongly dependent on initial conditions and limited parameter choice, the performance of the ABM proof-of-concept provided impetus for further development. In the near future, initial conditions and some parameter choices might be made on the basis of such real-time available data, on the basis that real time data can help govern the model and simulation and if available should be incorporated when possible.

Finally, the work added insight into the type of transient system behaviour captured by this model. After discussing data collection in the next chapter, the next versions of the ABM are presented; these are more suited to investigating what-if scenarios by virtue of the simulator being reconfigurable to an increasing degree without recompiling.

3.3 References

[1] G. Polya, Mathematics and Plausible Reasoning: Volume 1 Induction and Analogy in Mathematics. Princeton University Press, 1990.

[2] S. Mukhi and M. Laskowski, "Agent-based simulation of emergency departments with patient diversion," in Electronic Healthcare, D. Weerasinghe, Ed. Berlin: Springer, 2009, pp. 25-37.

[3] Kanagarajah, A. K., Lindsay, P. A., Miller, A. M. and Parker, D. W. (2006). An exploration into the uses of agent-based modeling to improve quality of health care. In: Y. Bar-Yam International Conference on Complex Systems, Boston, MA, 25-30 June (2006).

[4] Sanders, D., Mukhi, S., Laskowski, M., Khan, M., Podaima, B.W., and McLeod, R.D., "A Network-Enabled Platform for Reducing Hospital Emergency Room Waiting Times using an RFID Proximity Location System", 19th International Conference on Systems Engineering, Las Vegas, (2008).

[5] Geographical Intrusion Event Mapping System (GEMS), http://gems.ee.umanitoba.ca, [Online June (2008)]

[6] County of Marin, Ambulance Diversion Guidelines, http://www.co.marin.ca.us/depts/HH/main/ems/documents/Policies/5400.pdf [Online June 2008]

[7] Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, V.1 N.4, August 1993, p. 397-413.

[8] Qt4 http://www.trolltech.com [Online June (2008)]

[9] Patvivatsiri, L., "A simulation model for bioterrorism preparedness in an emergency room", http://www.informs-sim.org/wsc06papers/061.pdf [Online January (2008)]

[10] Doupe, M., Kozyrskyj, A., Soodeen, R., Derksen, S., Burchill, C., Huq, S., "An Initial Analysis of Emergency Departments and Urgent Care in Winnipeg" http://mchp-appserv.cpe.umanitoba.ca/deliverablesList.html, [Online (2008)]

[11] Canadian Triage and Acuity Scale, (This reference outlines Canadian Triage Scales illustrating the types of rules that can be embedded into the Agent Based Model) http://www.caep.ca/template.asp?id=B795164082374289BBD9C1C2BF4B8D32 , [Online (2008)].

[12] Wolfram, Stephen, *A New Kind of Science*. Wolfram Media, Inc., May 14, 2002. ISBN 1-57955-008-8

Chapter 4

Data Collection for Investigating "what-if" Scenarios.

A practical instantiation of the ABM-ML based decision support tool proposed in Chapter 1 will require high fidelity data as input, where the data is of higher fidelity and variety than has been considered in previous chapters. Such data would be used to develop the model's structure, agents, and their interaction; numerical simulation parameters; initial conditions; as well as direct input into related ML systems.

Specifically, this chapter addresses data collection and integration from diverse sources such as RFID-based Real-Time (patient) Location Systems (RTLS), wireless sensor networks, or a network enabled smartphone platform. The majority of this chapter is concerned with extensions to the original ABM itself, in order to model data collection using an RFID-based RTLS system. This also serves to underscore the evolution of the ABM into emerging areas of application, and from a descriptive tool (suitable for investigating general system dynamics) to more prescriptive capabilities (investigating the effects of system parameters, individually and in combination). Therefore, the ABM at this stage in development is demonstrating its potential as a tool to investigate "what-

if" scenarios and support policy decisions. Higher fidelity data, specific to the ED being modelled will be necessary to address "what-if" queries that are applicable to a specific real-life ED.

RTLS systems based on RFID or similar technology are expected to be deployed at an increasing number of institutions in the near future. Within the RTLS ABM framework presented, new opportunities to apply Machine Learning exist. For example, automated placement of RFID readers, potentially optimizing coverage for a particular budget or fixed number of readers. Such optimization would take into account emergent agent movement patterns, based on floorplan topography and patient flow throughout the treatment stream.

Towards the end of this chapter some results are presented, suggesting that it is possible to combine real-world disparate data sources, even when data is imperfect in nature, to gain some insight into the behaviour of a system. These considerations are a necessary first step when integrating collected data as input parameters to the ABM, or directly into a Machine Learning system. Finally, advantages are suggested that an ABM featuring visualization carries with respect to effectively utilizing healthcare practitioner expert knowledge of the system being modeled. The ABM approach serves as an excellent communication tool between healthcare practitioners and the modeller, aiding in model construction and validation. Practitioners communicate model requirements in their natural language, model parameters are negotiated and decided in a familiar lexicon, and the behaviour of the model closely resembles this description. It is a conjecture of this thesis that this leads to fewer systemic errors in the modeling process.

This author's contribution to the following research was the direction of the modeling efforts, coordinating and designing simulation experiments, and coding the core model upon which the RFID based RTLS model is based. This chapter is based on [1].

4.1 Uncertainties Inherent in RFID Tracking Systems in an ED

This chapter presents an Agent Based Modeling tool to assist in the deployment of RFID based tracking systems in healthcare facilities. The environment modeled here is an emergency department, with emphasis on patient tracking. The focus of the work is to quantify and assess the uncertainty and error associated with RFID tracking systems. The work extends the utility of RFID systems beyond asset and inventory control to patient tracking and highlights uncertainty as a critical issue in the data obtained via RFID tracking systems.

4.1.1 Introduction

Increasingly, healthcare management includes the implementation of technology to track the status and movement of various agents within the healthcare environment, including patients and physical assets. This is often a means of understanding patient flow, controlling inventory, tracking equipment usage, and thereby optimizing resources within the environment [2]. This chapter presents an Agent Based Model (ABM) of a hospital emergency department with extensions to modeling the provisioning of a real-time location system (RTLS) using radio frequency identification technology (RFID). The

work provides insights to the healthcare management professional into the degree of uncertainty one may anticipate when deploying a RFID based RTLS within a healthcare environment. The chapter is organized as follows: section 4.1.2 outlines the basics of RFID tracking and the concomitant uncertainty that is often overlooked when deploying an RFID system for asset and patient tracking. Section 4.1.3 briefly overviews Agent Based Modeling within a healthcare facility and specifically within an emergency department. Extensions of the healthcare model to incorporate RFID are also discussed. Section 4.1.4 outlines Agent Based Modeling simulation results used to evaluate a healthcare RFID RTLS. Data presented include the expected uncertainty as a function of reader placement. Section 4.1.5 provides a summary of the work and future opportunities.

4.1.2 RFID Systems

Radio frequency identification (RFID) systems have been available for several decades, and with technology advances they have recently become increasingly attractive alternatives in supply chain management for inventory control and asset tracking [3]. Less traditional RFID extensions have been proposed as solutions to tracking people, typically patients within a healthcare environment [4][5], in the interests of patient safety and patient care. As a consequence, a number of healthcare facilities will or have adopted RFID as their technology of choice and are evaluating its potential range of applications. The objective of this work is to assist healthcare managers and practitioners in appreciating the scope and limitations of RFID technology, and the inherent uncertainties

associated with any tracking system and – in this case – RFID, and ABM is the tool used to investigate this particular scope.

The evolving RFID RTLS are meant to augment and/or automate existing data capture (electronic records systems) in place in many healthcare institutions. A typical patient trajectory capture system may include the collection of time of arrival, time of triage, time and duration of treatment, and time of discharge from the emergency department. In some cases, data are entered by healthcare workers for multiple patients in aggregate when there is a break in the workflow. As such, there may be considerable uncertainty associated with the data themselves, making it difficult for policy-makers to make statistically verifiable decisions as they attempt to optimize patient access and flow through the healthcare facility. In this context, RFID systems offer a complementary and more automated means of data collection. However, critical issues are again associated with uncertainty and consequent inference required to remove ambiguities in the data.

A brief discussion follows as it relates to the model and simulations undertaken, although a complete technical overview of RFID specifications is beyond the scope of this chapter. RFID systems are tag-and-reader systems, comprised at minimum of a number of tags and at least one reader. RFID systems operate over a variety of frequency ranges and with a variety of complexities associated with the tags as well as readers. In general, a tag is associated with an asset or patient and the reader is associated with a location in the physical environment. The tag can be either passive or active, the latter requiring a battery allowing it to power a transponder and allowing it to be interrogated by a reader. A passive tag obtains its power from the field of the reader, allowing it to effectively “transmit” its identification back to the reader [6].

This simplified view of the electronics and processing already identifies several issues concerning uncertainty associated with the data. First, it should be noted that once a tag is read, it is primarily a proximity measure, meaning the tag is somewhere within the proximity of that reader. The low frequency passive tags (<20 MHz) use inductive coupling as opposed to propagating electromagnetic radiation, and as such are typically limited to fairly close proximities (~ 1-2 m). As a consequence, when attempting to create a patient trajectory through a healthcare facility, time-stamped historical data are required and rules are used to infer a patient's approximate location. Second, a particular reader and tag system will have time-varying non-isotropic capture areas, impeded by distance and other objects in the surrounding area. Third, in the case of the passive tag with virtually no processing power, a reader may read a tag at apparently random intervals or may not read the tag at all, depending on tag orientation and its environment. A fourth issue is interference, where closely-spaced readers interfere to the point where the interfering readers fail to read a nearby tag. The problematic case is the lack of a read (missed read) by one or both readers. As a simple example, of dealing with uncertainty, an estimate of queue lengths using RFID is presented in [7].

While both passive and active RFID tracking technology will continue to evolve and uncertainties in operation and quality will be mitigated, these uncertainties will likely never be completely eliminated. As such, modeling can play an important role in optimizing the implementation of a RFID RTLS patient tracking system. The next section overviews the simulation environment developed here for planning and managing a RFID RTLS system.

4.1.3 Agent Based Modeling

ABM has emerged as a simulation technique which attempts to model a system in as much detail as possible. This allows for interactions between agents to emerge, emulating the real world as closely as one is able. ABM is systems modeling, approached from the ground up or from the perspective of its constituent parts, in order to build an aggregate picture of the whole. Systems are modeled as a collection of agents, their individual behaviours, and their interactions. Agents are autonomous decision-making entities (generally, human beings, but can also include inanimate objects) able to assess their situation, make decisions, and compete with one another on the basis of a set of rules. ABM's conceptual depth is derived from its ability to model emergent behaviour that may be counterintuitive or, at minimum, its ability to discern a complex behavioural whole that is greater than the sum of its parts. ABM provides a natural description of a system that can be calibrated and validated by representative subject-matter experts, and is flexible enough to be tuned to high degrees of sensitivity in agent behaviours and interactions. ABMs are particularly well suited to system modeling in which agent behaviour is complex, non-linear, stochastic, and may exhibit memory or path-dependence [8][9].

A deterrent to the acceptance of ABM is that it is not a technique that lends itself to sensitivity analysis or the modeling of steady state phenomena without a potentially unrealistic number of simulations and substantial statistical analysis being undertaken. Although this position is generally valid, it is also the case that the advantages of ABM complement other techniques, and that it provides one of the most useful tools available

in terms of knowledge transfer and requirements capture, independent of whatever other techniques may be also employed. An ABM has a near perfect correspondence to the problem as understood by both the practitioner as well as the ABM implementer. In particular, there is a significant body of literature on mathematical modeling techniques applied to healthcare management, which can be complementary sources of data and cross-validation to ABM [9][10].

Many ABMs are developed to provide an opportunity to gain a better understanding of operations through the use of what-if scenarios. This is the approach used in this study. An ABM initially developed for improving patient access to healthcare [11] is extended to allow modeling a RFID RTLS augmented emergency department.

Chapter 3 demonstrated the basic model allows for various configurations, including provisioning the number of healthcare workers (HCW), the number and characteristics of patients, as well as the actual topography of the emergency department. Operational parameters in the emergency department include registration, triage, waiting and treatment areas. Currently, the model uses a prototypical layout which can be tailored to a specific facility. Of primary interest to a RFID RTLS is the accuracy and precision of patient trajectory through the emergency department. As each agent behaves in an autonomous fashion under the control of the simulator, an exact trajectory is recorded for each patient, which is then available for comparison and validation to trajectory data captured via the RFID system. A schematic of the ABM at an instance of time is illustrated in Figure 4.1. In the figure, patients and HCWs are illustrated, providing a visual animation as the simulation progresses. In effect, the animation is a tool used to

transfer information between practitioners and modelers, whereas the collection of data for statistical analysis proceeds without any type of visualization.

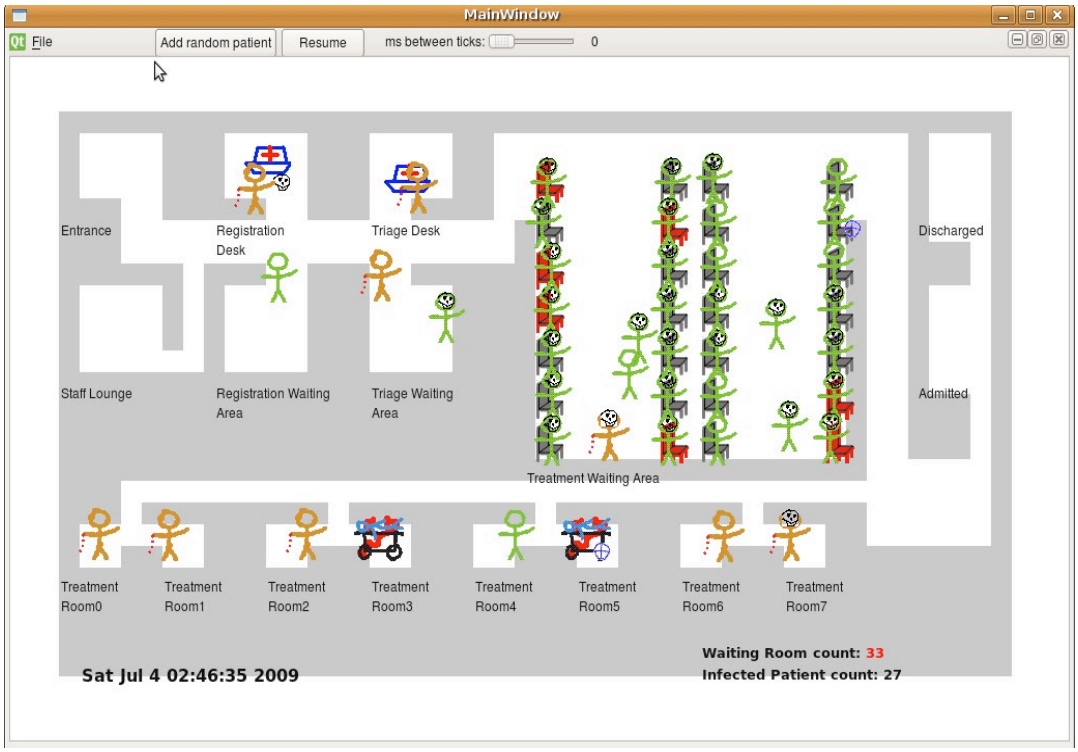

Figure 4.1. Emergency Department Layout

The RFID RTLS environment allows for the overlaying or placing of RFID readers which serve as inanimate agents. The readers can be placed by a design engineer manually or in a more systematic manner. For the purposes of this type of simulation, the former is used to allow for placement roughly based on the density of readers with a specified granularity. In practice, actual reader placement would be subject to access, mounting, and functional constraints, with read patterns also modified as a consequence of the local environment (walls, furniture, equipment, required clearances, as well as people, the latter being time-varying). Tag collision is not considered in this simulation;

it is assumed that over the duration of tags being within a reader's range, any colliding or interfering tags would be resolved in a time period considerably shorter than the tags traversing the reader. The resolution of the collision would be a consequence of multiple reads of the close proximity tags. This would also not be an issue with more sophisticated tag technology with collision avoidance.

For the RFID RLTS simulation, the exact trajectories of patients are compared with estimated trajectories inferred from RFID tag reads. Patient trajectories through the emergency department are largely governed by the triage score, by HCW resources, as well as by issues such as social distancing as aspects of agent behaviour. Four primary cases are considered: under-provisioning, two cases of intermediate-level provisioning, and over-provisioning of readers throughout the emergency department. The primary metric used for determining the quality of the RFID placement is the spatial error from the patients' actual position. The actual cost of readers or installation has not been considered, as the cost monotonically increases with an increased number of readers.

It is of primary importance that the RFID system be provisioned to minimize error with respect to patient tracking. Figures. 4.2 and 4.3 illustrate the model running with a placement of readers illustrated, highlighting a read of a patient tag as the patients traverses the emergency department.

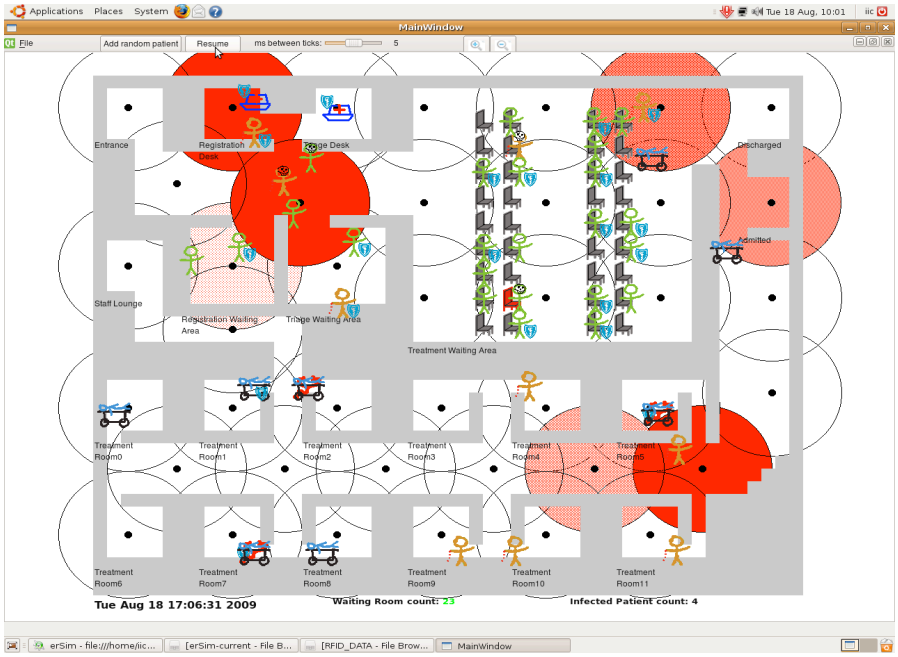

Figure 4.2. Snapshot of the Simulation at time t

In Figures 4.2 and 4.3, patients are moving between registration, triage, waiting and treatment rooms. The reader placement is seen as a grid of concentric dots and circles, with the radius indicating the reader range. The darkest colored circles indicate that a reader has just read a tag whereas the lighter colored circles indicate that a tag was read previously. Simulation parameters are set to resemble actual records of waiting times for emergency departments of similar resource. Although not exact, patient arrival rates and service times are adjusted to reflect times spent in emergency departments.

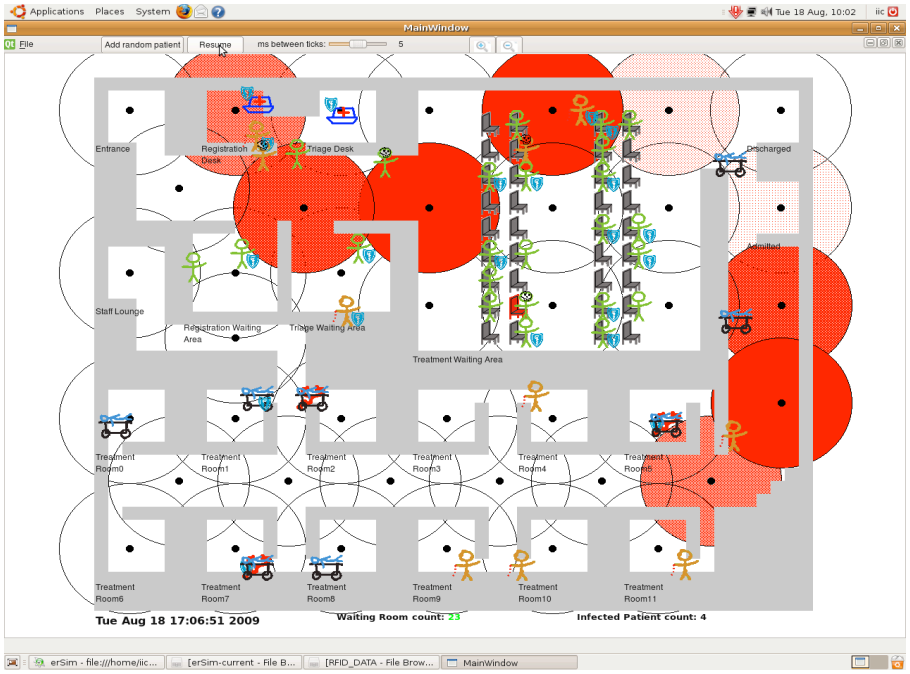

Figure 4.3. Snapshot of the simulation at time t + Δ

Typical simulation-generated values for waiting times are illustrated in Table 4.1. In effect, these are in arbitrary units, reflecting statistics generated over 500 simulation runs of a given set of parameters. Qualitatively, they are as expected with high priority patients waiting for minutes (simulated), while those with less severe ailments waiting for hours with more considerable variation. No policy was in effect that would allow for a fast track capacity trigger thereby fast tracking low priority patients to reduce the emergency department backlog.

4.1.4 Simulation Results

The measure used in this model to capture uncertainty or error is the trajectory difference inferred from the readers in contrast to a patient's actual position as known within the

simulation. As such, when a person is recorded within the range of a reader, the error would be zero. As the patient moves beyond the range of the reader, the error is the Euclidean distance from the reader to the actual patient position. Other measures would be equally reasonable at this point in the study and well within the modeling alternatives. Once a patient moves within the range of another reader, the error would again return to zero. A complication occurs when modeling interfering readers (an over-provisioned scenario). Here a dead zone is modeled, reflecting the overlap of two readers' ranges. The difference to the error measure is that the read may be delayed as an agent enters the reader range which was reduced by a dead zone, contributing a longer period of proximity uncertainty or error. For the purpose of the over-provisioned simulations here, the error was still considered to be zero when a patient was in a dead zone as he or she is still likely within the range of a close proximity reader.

The reader configurations are shown in Figure 4.4, illustrating the placement of five, ten, twenty, and forty readers manually placed throughout the emergency department. Readers are again indicated by black dots, with their ranges indicated by circles. Readers in the process of a read are highlighted. The placement scenarios are not optimal but are an attempt to capture patient flows through major traffic routes in the emergency department modeled.

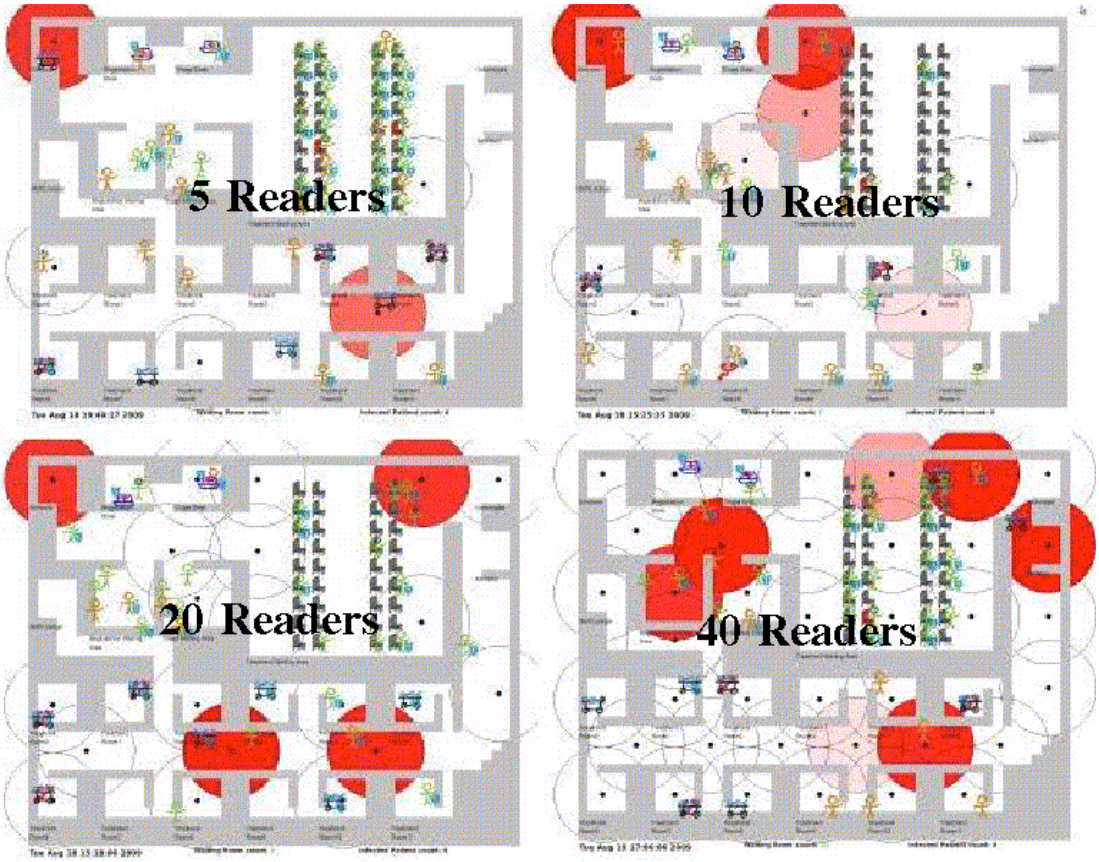

Figure 4.4. Location/Placement of five, ten, twenty and forty RFID reader scenarios.

Table 4.1 Typical Waiting Times for Low, Medium, High Triage Score Patients

Type of Patient	**Waiting Time (min)**	
	Average	***Standard deviation***
Low Priority (high triage score)	207	133
Medium Priority	94	30
High Priority (low triage score)	7	1

Figure 4.5 illustrates a typical error difference between reader-inferred location and actual location for a single patient, plotted over the duration of stay in the emergency department for four reader configurations. The trajectory error illustrated is for a similar reader arrangement as seen in Figure 4.4, illustrating the trajectory error for typical patient instances. The behavior of the patient is highly stochastic, governed by a number of random variables. In Figure 4.5, the error is plotted as being the Euclidean difference between the last read reader and the patient once outside the read range. When a patient is within a reader's range the error is set to zero.

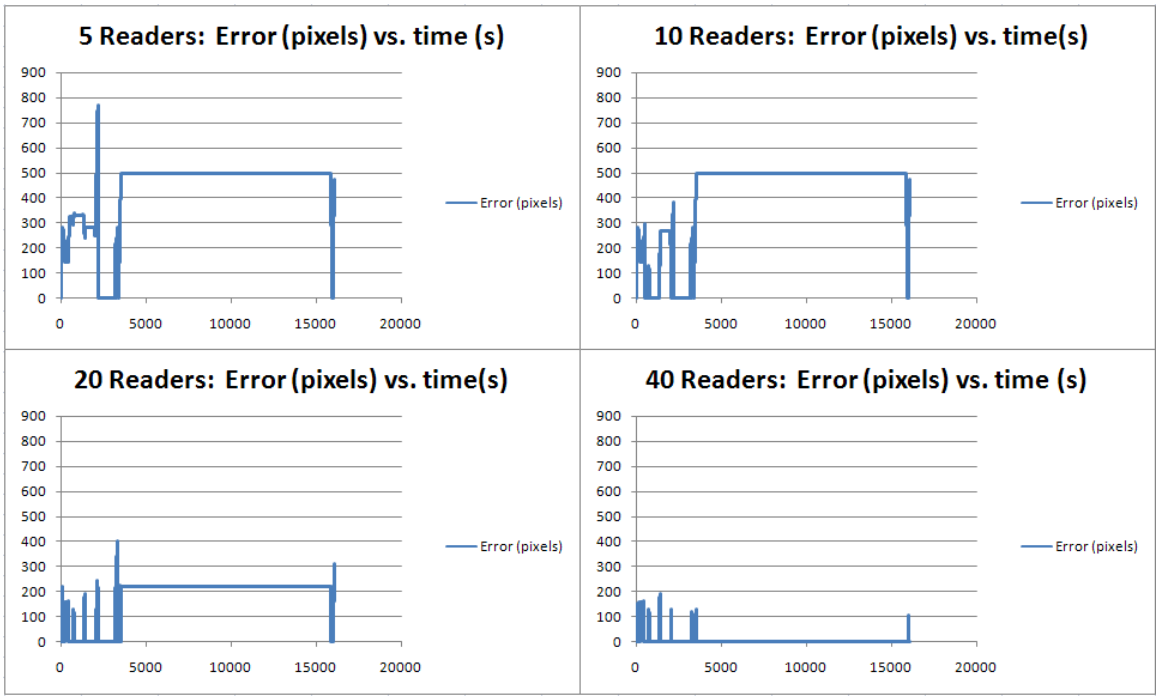

Figure 4.5. Location error between RFID and actual patient location (for one random patient, in each simulation)

Table 4.2 summarizes the normalized error for both spatial as well as temporal error. Temporal error is simply defined as the percentage of time that a patient is not in the

proximity of a reader. The simulation was run 500 times at each of the four reader configurations, with roughly 200 patients visiting the emergency department in each simulation. The layout is drawn on a 1024 by 800 pixel grid, corresponding to an area of approximately 30 by 24 m (100 by 80 ft). For example, in Table 4.2, the patient would be in the read range of a reader 1416/10326 or 14% of the time, with a normalized error of approximately 351 pixels or 11 m (35 ft). As the number of readers increase, the error or uncertainty is clearly reduced. It should be noted that although the uncertainty is decreased, there is a point of diminishing return. Although not linear, the cost of provisioning a 40 reader emergency department as well as its maintenance is significantly greater than an emergency department with 20 readers. In addition, any inference of behaviour as a consequence of the location of the tag being read is not considered. For example, if the last read reader was located in the waiting room, one can likely infer that the patient is still in the waiting area until another reader reads the patient's tag.

One of the most interesting aspects of the simulation is the percentage of time that a patient would be in range of a reader. From these experiments, in a system under provisioned, the patient's position may only be known approximately 14% of the time, whereas for the over provisioned case the patient's position is known approximately 95% of the time. Again it should be noted that reader placement should be also governed by event triggers such as a patient being brought to a treatment room correlated to the event of the HCW also being present. As such, one can live with a high degree of uncertainty in certain areas less in others. Clearly, one conclusion which can be drawn here is that new sources of data bring with them new opportunities to introduce uncertainty.

Table 4.2 RFID Location Errors (Temporal and Spatial)

Readers	Total Simulation Time (s)	Time which Patient Location is Known (s) (%)	Accumulated Spatial Error (Pixels)	Normalized Spatial Error (Pixels)
5	10326	1416(13.7%)	3836772	351
10	11226	3405 (30.0%)	2405351	217
20	10274	5616 (54.7%)	927877	86
40	11833	11198 (94.6%)	68839	6

4.1.5 Conclusion

This chapter presented an ABM useful in the design of an RFID RTLS for tracking patients in an emergency department or similar healthcare setting. RFID based RTLS systems hold great potential to play a significant role in augmenting healthcare electronic records. However, they are not without their limitations. One of the more challenging obstacles to overcome is dealing with the inherent uncertainty of relatively low cost RFID systems, which requires some backend processing to limit the uncertainty. As RFID systems continue evolve and more sophisticated tags are deployed, it is reasonable to expect additional levels of physical layer triangulation be employed, as well as more intelligent protocols that may also query tags in an attempt to yet further reduce location estimation error [12] . Many of these issues will be resolved in evolving medium access control protocols as well as at the application layer for the RFID RTLS system.

Nonetheless, this chapter illustrates the role that ABM has in provisioning a RFID based tracking system. Model extensions should also include site surveys that can be included in the simulation model.

4.2 Discussion

4.2.1 Data Collection Using Mobile Devices

Appendix A [13] presents an example of another agent (patient) tracking technology, this time using commercial smartphone or other mobile devices as the means of data collection. The smartphones would either be given out as part of a research effort or volunteers could offer to run the tracking software in the background on their own compatible devices. Using some custom software to manage connectivity, the tracking software periodically reports to a central server, for example the CORE server mentioned in Chapter 3. These periodic reports would contain the smartphone's location provided via GPS, as well as other information such as identifying other nearby smartphones running the same tracking software. This method of data collection would be more suitable to track the ambulances from Chapter 3 or infer patterns of patient movement from home/work to hospitals, rather than patients within a hospital. It should be noted that only the nature of backend processing would need to be changed from the presented framework in order to perform tracking and analysis of patient and ambulance movement suggested in Chapter 3. It is likely that in the case of metropolitan area patient tracking, a subset of all patients would be carrying tracking devices, and some inference would have

to be conducted in order to gain some understanding of the behaviour of the patient population as a whole. In the case of ambulance tracking, installation of tracking systems could be mandated. This is becoming increasingly common, referred to colloquially as "lojacking" (opposite of *hi*jacking as these tracking systems are used to counter vehicle theft).

4.2.2 Data Fusion

Appendix B [7] discusses deployment and subsequent data inference performed with respect to an experimental deployment of an RFID tracking system, outside of a healthcare setting. The experiment successfully fused queuing data from the RFID readers with data extracted by a human observer from a webcam recording of the queue. This demonstrates how useful features can be extracted, even when the data is intermittently available, noisy, and incomplete to a considerable degree.

A deployment of RFID tracking for healthcare is expected to be far better provisioned than the experimental setup presented in Appendix B. Rather than fusing RFID tracking data with data obtained from video, an improved system would combine RFID data with increasingly detailed electronic Emergency Department Information System (EDIS) information.

4.2.3 Working With Practitioners

Appendix C elucidates the advantages an ABM decision support tool with visualisation offers with respect to communication between the modeller and practitioners (healthcare workers, and administrators). This aspect is of importance because the modeller is not typically an expert in the domain being modelled. Throughout the course of requirements gathering, one such advantage is that by its very nature, the model will closely resemble the natural description of the system provided by practitioners. This aspect is also useful when working with practitioners towards validating the model, as well as communicating results. Furthermore, by reducing the number of steps at which knowledge translation and abstractions are made in the course of model construction, the opportunities to introduce errors or misinterpretations are reduced.

4.3 Summary of Chapter 4

The work in this chapter supports the conjecture that in order for an ABM based decision support tool and any associated ML systems to address "what-if" scenarios as part of a policy decision-making process within a real hospital, real-life data will need to be collected and integrated with the ABM. This data would be used for parameter choices, initial conditions for simulation, and as input for an associated ML system. In other words, real data collected from bio-surveillance methods are one piece of the puzzle for developing a successful ABM-ML based decision support tool for healthcare policy.

The work described in this chapter sets the stage for integrating real time patient tracking data by illustrating the sort of low-level data that is expected to be available in

the near future. The ABM framework, enables modeling the collection of real time patient location data – thusly providing a sort of closed loop feedback for deployment of real time patient location technology.

Extensions were made to the ABM allowing it to model an RFID based RTLS system. This both represents progress towards the deployment of such a system, as well as illustrating further development of the ABM into novel application areas. The RTLS model itself suggest further ML applications, for example optimization of the simulated RTLS deployment.

A complementary mode of patient and ambulance tracking using smartphone platform such as Android is briefly discussed. Then, some encouraging results were presented, relating to the feasibility of extracting parameters suitable for the ABM from multiple sources of raw data. Namely, improved EDIS data may include higher fidelity patient arrival rates, capturing more detail about the condition of arriving patients, as well as arrival rate variance by time of day, day of week, and seasonal variation. EDIS data could provide more detail on the amount of time patients spend involved in various ED processes, specifically doctor treatment time but also extending the model to capture other ED processes such as laboratory-based patient diagnostic tests. Anticipated future patient RTLS implementations, such as the proposed RFID-based or mobile platforms such as Android would augment and fill gaps in EDIS data, especially regarding which patients and healthcare workers were in contact and for how long. This contact information is especially useful when modeling infection spread, which will be discussed in Chapters 6 and 7. The Android platform has the added potential to track patients

outside the healthcare facility for modeling city wide scenarios, again, especially with respect to modeling infection spread.

Finally, this chapter underscores the role ABM plays as a communication tool between healthcare practitioners and modellers, facilitating model design and validation by virtue of a common natural language description translated into a model that closely resembles that description.

4.4 References

[1] M. Laskowski, B. Demianyk, M.R. Friesen and R.D. McLeod, “Modeling an RFID tracking system in an emergency department”, presented at the *IEEE Workshop on Healthcare Management*, Venice, Italy, 2010.

[2] S.W. Wang, W.H. Chen, C.S. Ong, L. Liu and Y.W. Chuang, "RFID application in hospitals: A case study on a demonstration RFID project in a Taiwan Hospital," *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, 2006.

[3] K. Michael and L. McCathie, "The pros and cons of RFID in supply chain management", *ICMB 2005 International Conference on Mobile Business*, pp. 623-629, 11-13 July 2005

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1493672&isnumber=32116.

[4] A.M. Wicks, J. K. Visich, and S. Li, “Radio frequency identification applications in hospital environments”, Hospital Topics, vol. 84, no. 3, pp. 3 - 9, 2006.

[5] J. A. Stankovic, Q. Cao, T. Doan, L. Fang, Z. He, R. Kiran, S. Lin, S. Son, R. Stoleru, and A. Wood, “Wireless sensor networks for in-home healthcare: Potential and challenges,” in High Confidence Medical Device Software and Systems (HCMDSS) Workshop, Philadelphia, PA, June 2-3, 2005.

[6] K. Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, 2nd ed., John Wiley & Sons, Inc., 2003.

[7] D. Sanders, S. Mukhi, M. Laskowski, M. Khan, B.W. Podaima, and R.D. McLeod, “A network-enabled platform for reducing hospital emergency room waiting times using

an RFID proximity location system," presented at 19th Int. Conf. on Syst. Eng., Las Vegas, NV, 2008.

[8] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," Proceedings of the National Academy of Science [Online]. 99(Suppl 3), pp. 7280-7287, 2002. Available: http://www.pnas.org/content/99/suppl.3/7280.full#xref-ref-3-1

[9] A. Skvortsov, R. Connell, P. Dawson, and R. Gailis, "Epidemic modelling: Validation of agent-based simulation by using simple mathematical models," MODSIM International Congress on Modelling and Simulation, pp. 657-662, 2007.

[10] L. Temime, G. Hejblum, M. Setbon, and A.J. Valleron, "Review article: The rising impact of mathematical modelling in epidemiology: Antibiotic resistance research as a case study," Epidemiological Infection, vol. 136, pp. 289-298, 2008.

[11] S. Mukhi and M. Laskowski, "Agent-based simulation of emergency departments with patient diversion," in Electronic Healthcare, D. Weerasinghe, Ed. Berlin: Springer, 2009, pp. 25-37.

[12] N. C. Nierenberg and D. Caliri, "Wireless tracking system and method utilizing tags with variable power level", U.S. Patent, 7504928, issued March 17, 2009.

[13] M. Laskowski, J. Allen, K. Ferens, M.R. Friesen, and R.D. McLeod, "Rapid prototyping vehicle-to-infrastructure applications using the Android™," presented at *The 9th International Conference on ITS Telecommunications*, Lille, France, 2009.

Chapter 5

Models for Decision Support

In this chapter, ABM is contrasted with complementary modeling methods, specifically, queuing models. Most notably, the ABM focuses on transient phenomena in contrast to the queuing model's focus on steady-state behaviour. More importantly, in the greater context of developing an ABM-based decision support tool for healthcare, this chapter introduces specific ways in which ML could be used to augment the ABM. The ML-augmented application, in this instance, is forecasting patient wait times within an emergency department, as an example of predictive data mining. An ML-augmented agent policy evolution task is also introduced, and is more fully developed in Chapter 7.

At this stage of development, the ABM is capable of addressing a number of what-if scenarios owing to an increased number of run-time specifiable parameters. The ability to obtain new what-if scenario results is exercised in the next chapter. Instead, the related queuing model presented here is used to perform a what-if comparison of a pre-emption versus non-pre-emption treatment policy. The utility of incorporating enhanced data sources such as those discussed in Chapter 4 is further supported throughout this chapter.

The remainder of this chapter is based on an article that appears in the journal PLoS One, "Models of Emergency Departments for Reducing Patient Waiting Times" [1]. This author's contribution to the following chapter is the design and implementation of the ABM, carrying out the ABM related simulations, as well as development of the combined ABM and ML framework presented. Although edited, there is significant overlap between the ABM discussion in this chapter and that of chapter three. The material is reiterated here for the purposes of side-by-side comparison with the Queuing Model and facilitating the discussion on applications of ML within ABM-ML decision support tools, which constitutes the main contribution of this chapter.

5.1 Models of Emergency Departments for Reducing Patient Wait Times

In this chapter, both agent-based models and queuing models are applied to investigate patient access and patient flow through emergency departments. The objective of this work is to gain insights into the comparative contributions and limitations of these complementary techniques, in their ability to contribute empirical input into healthcare policy and practice guidelines. The models were developed independently, with a view to compare their suitability to emergency department simulation. The current models implement relatively simple general scenarios, and rely on a combination of simulated and real data to simulate patient flow in a single emergency department or in multiple interacting emergency departments. In addition, several concepts from telecommunications engineering are translated into this modeling context. The

framework of multiple-priority queue systems and the Genetic Programming paradigm of evolutionary Machine Learning are applied as a means of forecasting patient wait times and as a means of evolving healthcare policy, respectively. The models' utility lies in their ability to provide qualitative insights into the relative sensitivities and impacts of model input parameters, to illuminate scenarios worthy of more complex investigation, and to iteratively validate the models as they continue to be refined and extended. This chapter discusses future efforts to refine, extend, and validate the models with more data and real data relative to physical (spatial – topographical) and social inputs (staffing, patient care models, etc.). Real data obtained through proximity location and tracking system technologies is one example discussed.

5.1.1 Scope

Hospitals represent a promising area where modeling and simulation can be effective tools in evaluating patient access and patient care policies and efficiencies. In many cases, the operations of an emergency department (ED) are over taxed, as they represent the necessary compromises between competing priorities. Although policies and practices evolve over time and best efforts are made to reduce patient wait times and other patient care parameters, often there is little quantitative analysis or feedback in the process, effectively described as "best intent open loop."

In this chapter, both agent-based model (ABM) and queuing model (QM) techniques are applied to the operations of an ED, specifically with respect to patient access and patient flow through the ED. The objective of this work is to gain insights into the comparative contributions and limitations of each respective technique. The broader

objective of the work is to contribute empirical input into healthcare policy and practice guidelines related to patient access and patient flow. To date, the work has generated general models (ABM and QM) relative to patient access and patient flow in EDs. These are currently built on relatively simple models of the physical layouts and social processes within EDs. Although derived from input from healthcare experts, the models represent low-level, coarse-grained models of EDs, as these are a suitable starting point from which to evaluate the model's validity. These general models are presented in this chapter, and they provide an opportunity (within and between the ABM and QM models) to investigate the relative sensitivities and impacts of various model parameters on patient access and patient care indicators.

In general, the ABM approach is applied in this work to investigate scenarios for resource optimization within the operations of an ED (for example, staffing scenarios). The QM approach facilitates quantitative analysis of operational parameters in EDs (for example, wait times). However, in both cases, the intent is to carry out predictive modeling with increasingly empirical inputs, which not only provide greater and more complex insights into the operations of EDs, but feed into the improvement of the ABMs and QMs themselves. To that end, this chapter discusses the opportunities and future efforts to refine, extend, and validate the models with more data and real data (vs. simulated data). Future opportunities and efforts will also focus on refining, extending and validating models with a more complex range of physical (spatial – topographical) and social models (staffing, patient care models, etc.), such as those extracted from real time location systems and emergency department information systems, respectively. Augmenting the range of agent behaviours and interactions in an ABM is an

interdisciplinary enterprise, and future efforts will rely heavily on input from healthcare experts.

5.1.2 Background

A considerable focus of the applications of ABMs has been on community-level epidemic modeling in human populations (see, for example [2] and [3]), as this is an important public health and policy issue with far-ranging health and economic impacts. Within healthcare settings, a literature exists with respect to applying ABMs, alone or in complement to other techniques, to the operations of EDs. In general, this literature addresses system-level performance dynamics, quantified in terms of patient safety [4], economic indicators [4][5], staff workload and scheduling [6][7], and patient flows [8][9]. While this literature addresses system-level operational concerns during periods of typical operation or stasis, there is also a literature on modeling of healthcare operations during critical incidents like disease outbreaks and terrorist attacks [10][11][12]. However, relatively little work exists in applying ABMs to healthcare policy development [13]. The authors' prior work includes both the development of a large scale (community-level) agent-based epidemic model [14], and more recently, an ABM for hospital acquired infections [15]. Figure 5.1 was previously presented in Chapter 1, but appears here again to position ABM in relation to Queuing Models for the purposes of comparison in this chapter.

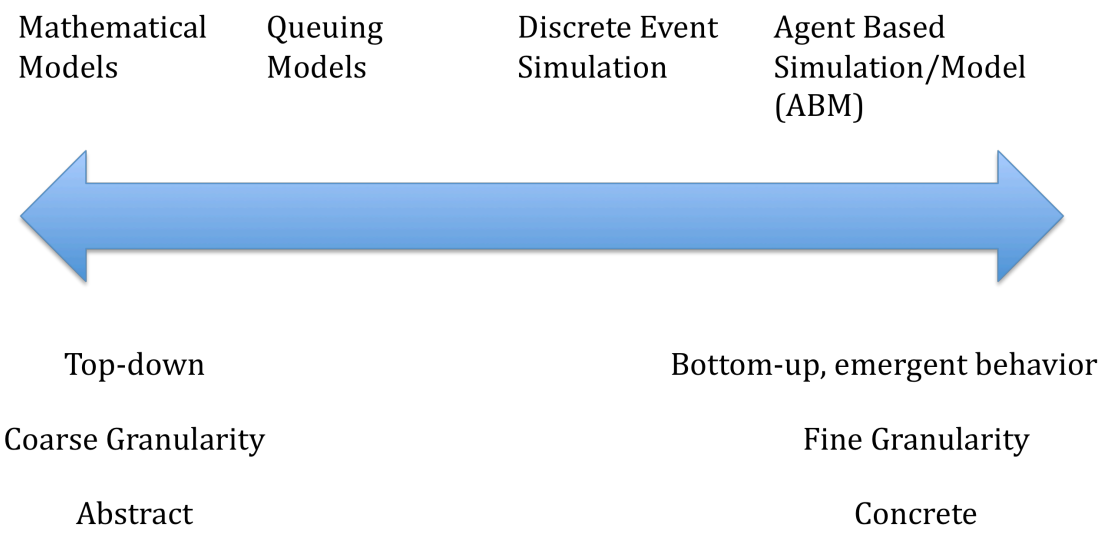

Figure 5.1. ABM in relation to other modeling methodologies

Complementary to ABMs, queuing-based modeling represents a well established and vetted methodology in operations research, with extensive applications in the service industries. Even though its application to healthcare is not new, this application has grown more recently, and the need is more recognized. For example, the CORS/INFORMS 2009 conference in Toronto, Canada devoted 15 sessions to healthcare applications, of which at least five are focussed on the applications of queuing theory. A recent issue of the flagship journals of INFORMS, Operations Research (Vol. 56(6)), was a special issue on Operations Research in Health Care. There is a clear recognition that QM can be applied creatively to understand and estimate the expected performance of the service processes in a healthcare system. Applications of QM in healthcare range from using it to study flows in EDs [16], analysis of delays for medical appointments [17], and for determining hospital bed requirements and allocations [18]. Most importantly, the model can be used to assess the operations of the healthcare system under different

'design' scenarios. By identifying the service points in a healthcare system, the associated topographic linkages between these points and the stochastic processes that characterize the arrival process of patients and service processes of healthcare staff, one can apply a QM to quantitatively describe patient flow through the systems as well as waiting times in the system. QMs allow us to assess different configurations of service nodes and triage rules. Most of the existing models for healthcare are strictly queuing models.

In addition, technologies are emerging that can be leveraged by hospitals to improve patient care. Two of the more obvious technologies and applications include intra-hospital tracking and internetworking. These technologies can allow for a more distributed approach to managing a number of interacting EDs. This is incorporated into one of the ABM applications described in this chapter, relative to evaluating ambulance redirection or other patient diversion policies. Previous work by the author presented a specific emergency department data collection application and architecture and extended it to a wide area Hospital/ED/Ambulance and patient diversion framework [19].

5.1.3 Basic ABM Framework

As per the ABM framework outlined in earlier chapters, this work focussed on an object oriented (OO), open-source visual simulator which can be used to gather data from a patient flow monitor information, applied to analyzing and forecasting patient waiting times. A screenshot of the simulator window at this stage of the research undertaken here, is shown in Figure 5.2.

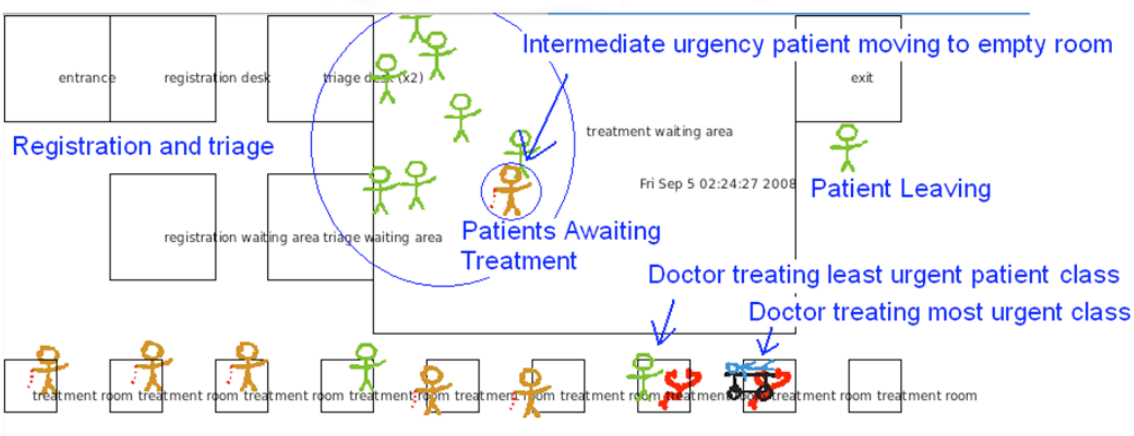

Figure 5.2. Screen Capture of the Basic ABM Simulator

Further details of the ABM simulator are presented in [19]. The OO paradigm allows for instantiation of EDs and allows for communication between EDs. In an actual healthcare setting, patient information could be effectively conveyed on dashboards within the participating EDs, as well as on a dashboard at a more centralized control location. In the current model, patient arrival rates and service times are based on estimates obtained by other researchers [21]. Further work will focus on driving the simulations on real world data collected in real time, such as that proposed in [22], where RFID or other proximity location system data would be used to augment the simulator. Further, the current model is able to place functional areas of the ED at arbitrary locations. Future work will integrate real spatial–topographical data taken from floor plans of the EDs wish to simulate. Floor plans from various EDs are readily available, such as the Halifax QE2 emergency department available at [23]. This type of topographical information is becoming more readily available and extremely useful for modeling purposes.

5.1.4 Basic Analytic Queuing Model

A further aspect of the work is to implement queuing models (QMs) as a complementary technique to ABM, as a means of gaining complementary, comparative, and high level insights into afore-mentioned healthcare applications. In contrast to simulation, a QM is analytic, able to provide insight expediently but often in exchange for accuracy. The current application was to develop a general baseline model of an ED suitable for comparison with the ABM simulation. Similarly, multiple EDs can be modeled as a network of queues, augmented with numerical techniques and assumptions of dependencies. The work fits into the overall objective of combining the ABM with the analytic QM in a hybrid that would be both fast and accurate. As the work develops, it may become evident that one approach is preferable, more applicable, or more insightful than the other, depending on the results desired.

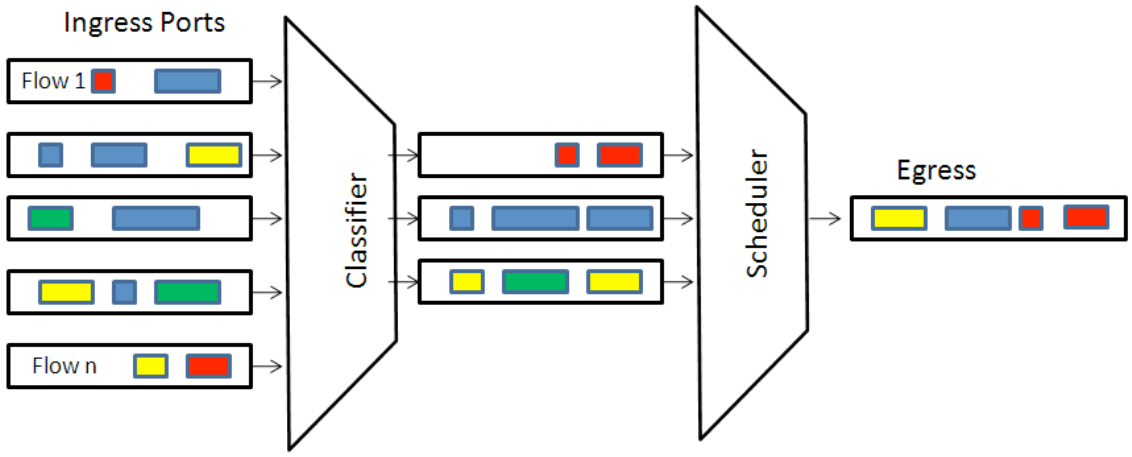

Figure 5.3. Queues within Telecom Equipment

In applying QMs to healthcare applications, the telecom analogue remains very applicable, as there is both a vast literature on simulation as well as on QMs. In addition, networks are topographically somewhat similar to a network of EDs and similar to flows within an individual ED. The following example could represent part of the patient flow through an ED, framed within a QM: a patient would arrive with an injury, be registered, triaged, scheduled for diagnostic services and treatment, and discharged. Figure 5.3 illustrates an analogous and familiar queuing situation in a telecom context. A point to note is that although QMs have been used to improve and analyze a wide variety of processes, the telecom field is one with close correspondences to many healthcare scenarios. For example, there are queuing phenomena common in telecom networks that decrease system performance that would have a corresponding analogue in healthcare, such as head-of-the-line blocking. Extending this notion, an analysis of algorithms applied in telecom networks to optimize system performance may also have novel applications to healthcare. An example of head of the line blocking in a healthcare setting may be a patient ready for discharge from the ED, but waiting for an as-yet unavailable bed on a ward.

Complexities are added to the model in that the arrival rates of heterogeneous patients are not governed by well-behaved statistics. In addition, the queues may be pre-emptive, in that if a person arriving with a serious injury would pre-empt others waiting in various queues. Available ED resources, including physical resources (beds, equipment, etc.) and human resources (nursing staff, diagnostic staff, physician staff, etc.) further add complexity to the model when compared to a communication or data network. However, some of the basic and overall performance measures are similar. For instance, the total

service time of a patient in the system (entry to exit) is a measure of interest in both a data network as well as an ED. In a network, analogous policies such as prioritization are used to prevent unbounded delay (time spend in the ED) from occurring for important traffic (more serious patients).

In terms of providing information to healthcare staff and administrators, a queuing figure provides an inherently familiar visual means of displaying bottlenecks in an ED. In general, patients in individual queues are often in a common waiting area. A dashboard display in the ED illustrating the various queues in real time could be a valuable means of displaying a snapshot of what is going on in terms of patient flows, routing, and delays.

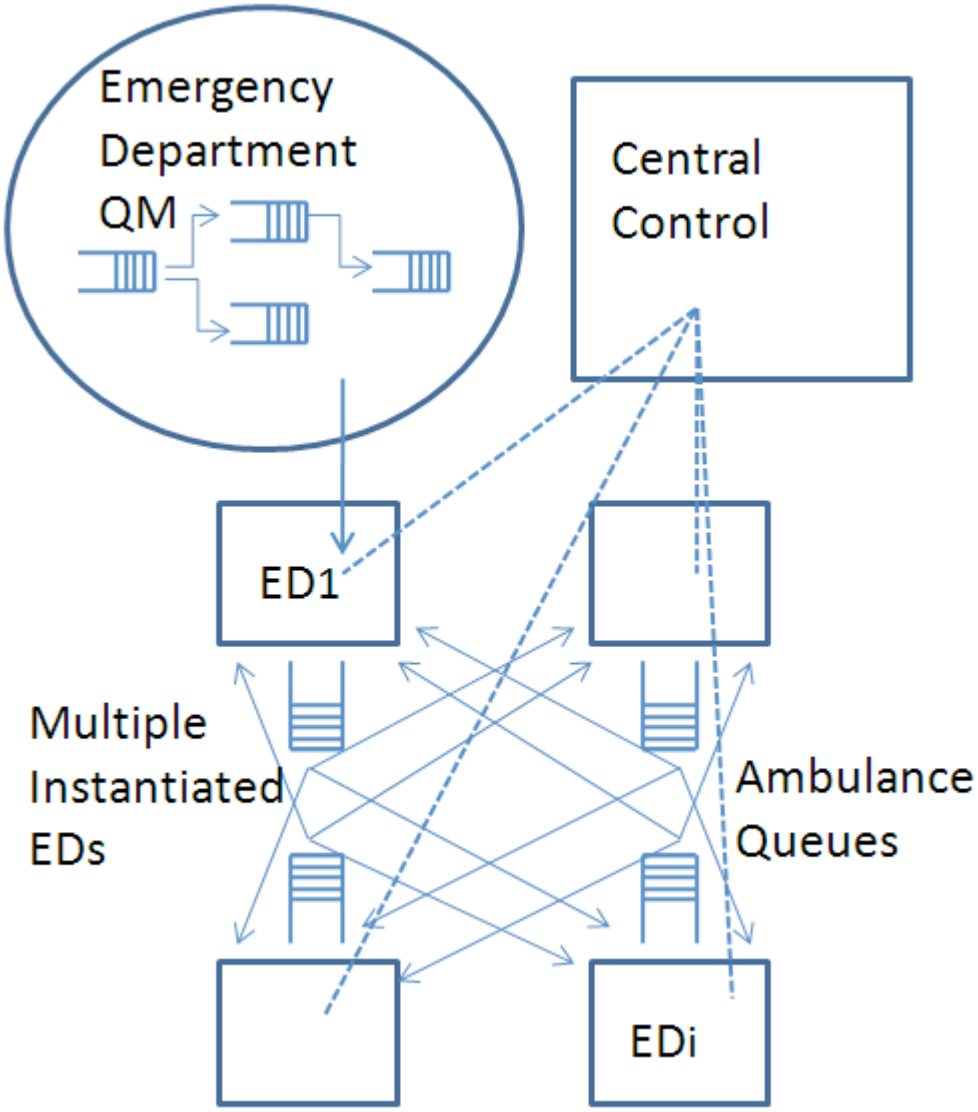

Figure 5.4. A Network of Emergency Departments Connected by Ambulance Queues

As an extension to a single ED modeled within a QM framework, an inter-hospital network of EDs will more closely resemble a complete graph, as in practice any ED could redirect patients to any other ED (Figure 5.4). In reality, there would also be a hierarchy of EDs, as some may be regionally designated trauma centres and/or priority centres for specific types of presenting injuries. In addition, geography may make it

more practical to divert patients to closer EDs, as more distant EDs may add to a patient's overall delay or time spent in the system. The complete graph of a network of EDs contrasts modern telecommunications networks in that communication networks are sparse graphs relaying packets of data as they traverse the network. While a modern telecommunication network does not closely resemble a network of EDs, the various services on a modern network do. However, a decided advantage of modeling multiple EDs as opposed to a telecom network is the feature of central control. Computer communication networks lack centralized control (although traditional telephony networks rely on control in establishing a path through a network on which an actual call can take place).

To address some of the complexities of networked inter-hospital QMs, a degree of simplification can be achieved by focusing the model on patient diversion for high priority patients. This approach would accurately model the high priority queues within EDs, with all non priority patients representing background noise in the system. Information required by the patient diversion scheduler in a coarse-grained approach to this scenario would include the patient triage level, the estimated delay at the initial receiving ED, and estimated transport delay and estimates of delay at the target ED. By necessity, the system would be a non-preemptive priority queue, in that, once a lower priority patient is in transit they would not be pre-empted in transit.

A variation on the above scenario which reduces the degree of simplification is to explicitly simulate the high priority patients and aggregate all other patient flow. As such, the explicit, high priority patients, as well as various policies and protocols would be modeled in detail. Modeling would not keep track of individual patients, but only

their aggregate impact on delay of high priority patients at individual EDs. In this scenario, the aggregated (background) patient traffic would be context (ED) specific.

A final extension to this work is to consider a network of EDs with some degree of hierarchy, based on ED capacities, priorities, and capabilities. In this case, patient diversion would not only consider queue lengths at various EDs, but also the priority level of patients. One may see, for example, the diversion of a less critical patient to a hospital with fewer resources, rather than contributing to a queue of low priority patients at a regional trauma centre.

5.1.5 ABMs for Patient Access to Emergency Departments

The current model is representative of a simple framework, suitable for simulations that provide insight into the relative sensitivities and impacts of the simulation parameters without necessarily quantifying them. As well, the current model can be validated on an ongoing basis, before and concurrent with adding requisite complexities.

The same basic ED patient flow model as was presented in Chapter 3 is used again here. A summary of this model is presented again in Figure 5.5 because it is illustrative in the following discussion. This model is similar to many multiple priority queue systems found in many telecommunication technologies, such as 802.11e [24]. 802.11e proposes wi-fi extensions for improving the more common 802.11 standard, through the use of priority queues serving various classes of traffic. When applied to healthcare, the utility of using analogues from telecom as a benchmark or reference is that these amendments are usually well vetted and studied and can be readily leveraged.

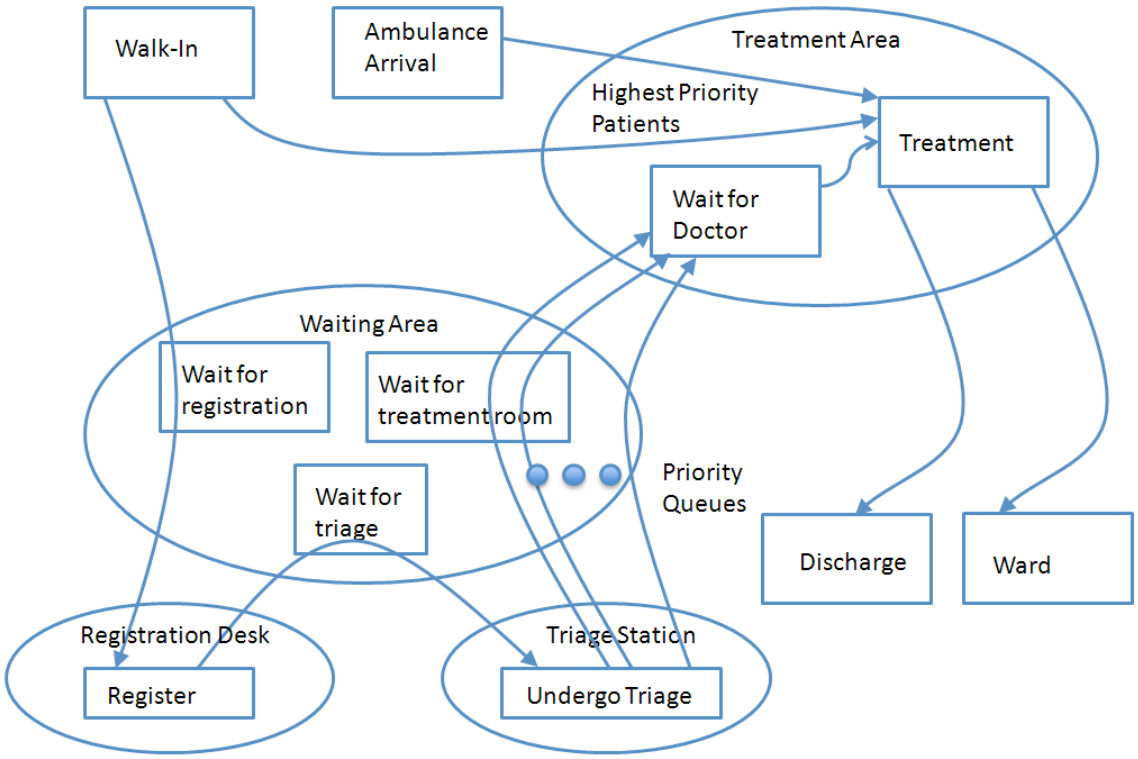

Figure 5.5. Model of Emergency Department Patient Service.

Model parameters include (but are not limited to) the number and types of agents, the range of agent behaviours, the range of agent interactions, the spatial-topographical nature of the environment, and the nature of the processes being simulated. In building an ABM from the ground up, the initial simplicities are necessary to validate the model qualitatively and on an ongoing basis as it is refined and extended. By its very nature, further efforts in ABM development focus on expanding the range and nature of model parameters to better reflect real-life environments and social processes, and this applies to the range of work described in this chapter.

5.1.6 ABM Simulation of Staff Allocation

A series of simulations was carried out relative to staff allocation in an ED, to investigate the utility of the ABM framework for optimizing resources and making informed policy decisions. The first scenario investigated, while simple, illustrates the effects of changing staffing levels, using multiple performance metrics.

In this scenario, basic ED scenario described earlier was simulated, with Triage Classes, Service Times, and Patient Arrival rates based on [21]. Three different staffing scenarios of two, three, and four doctors working in the ED were compared. The simulation was allowed a "warm-up" period of 24 hours, then observations were made during the following 24 hours. Ten independent trials were run; average treatment queue length is shown in Figure 5.6. Alternatively, doctor utilization or individual patient waiting times can also be instrumented. The staffing simulation is qualitative, but represents one instance of this type of model that can be investigated at individual hospitals.

Intuitively, the simulation results appear reasonable. Figure 5.6 shows the average number of patients waiting for treatment as a function of time (in seconds). The value of the results at this stage of model development is qualitative: the ED model staffed with two doctors is understaffed, evidenced by a continually increasingly patient queue. At the other end of the continuum, the ED staffed with four doctors results in a patient queue of near-zero; however, corresponding results indicate that the doctors are underutilized. This allows discussions to occur relative to resource allocation and optimization, in this case, physician resources.

In refining the simulation, one would seek to apply context-specific patient, staff, and patient care parameters, as discussed earlier. Where individual EDs are instances of a regional hospital authority, further extensions of the work are to model multiple facilities and thereby provide a means of assessing patient diversion policy between facilities.

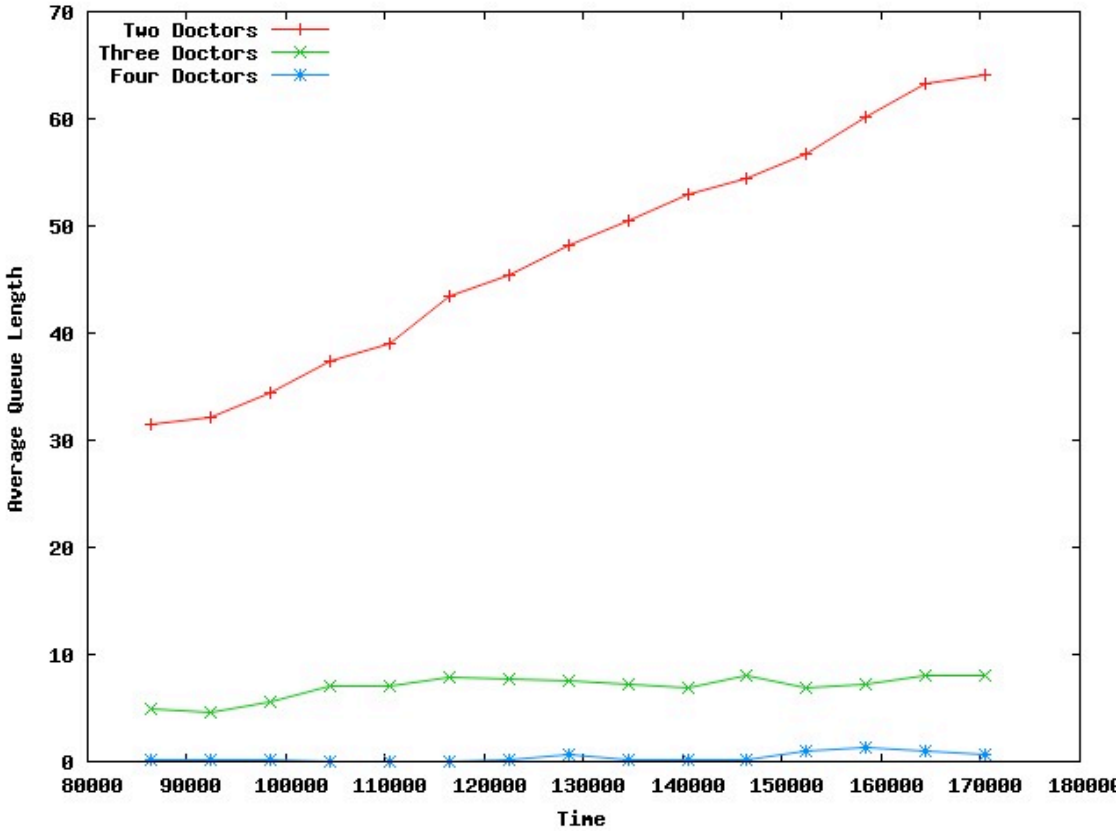

Figure 5.6. Average Queue Lengths for Varying Number of ED Doctors on Duty

Prediction based on modeling and simulation is extremely difficult and potentially error prone. Confidence can be enhanced as the system is in operation and predictions tracked. The conjecture made here is that the model of individual or interacting emergency departments, augmented with whatever available empirical data is available,

would still be preferable over loop policy decision making. Sensitivity analysis associated with both the ABM engine (numerical stability) as well as validating the null hypothesis in term of a policy's effectiveness is still required.

In refining the simulation, one would seek context-specific patient arrival rates. For these purposes, the individual EDs are instances of a regional hospital authority; this facilitates the opportunity to model multiple facilities and thereby provide a means of assessing patient diversion policy between facilities.

5.1.7 ABMs for Inter-hospital Patient Diversion

While the previous sections focussed on ABM simulation of patient access and patient care within EDs, this section provides an overview of how a modeling system could be extended between hospitals and integrated within a regional health authority informatics system. The discussion is model-agnostic, but for discussion purposes, an ABM is presented. An ABM of an ED is useful on its own [4], and its utility can be enhanced when combined with real data captured via tracking technologies and networking capabilities. In this specific application, the ABM is a distributed model across a number of regional hospitals, with an emphasis on utilizing data collected and analyzed in near real time. A novel aspect of the present model is the use of congestion avoidance algorithms from telecommunications engineering redeployed as a model for evaluating patient diversion policies. Again, the current models would benefit from the addition of real data in near-real time; such data is becoming increasingly available, although in some instances it may have to be inferred or estimated [22]. Subsequently, this data needs to be shared among regional hospital and health care facilities. Availability of and

access to data are both technical and political challenges, although they are optimistically considered to be surmountable.

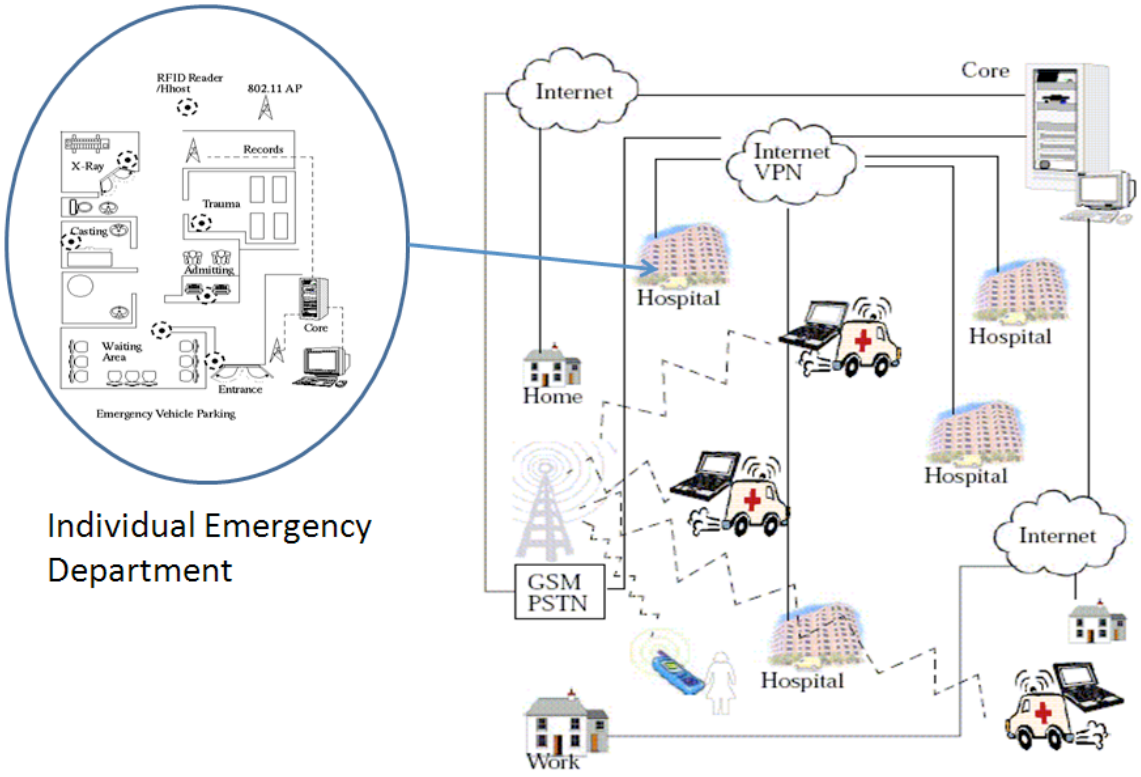

Figure 5.7. Wide Area Deployment of the Framework Illustrating the Major Stakeholders or Agents

Figure 5.7 illustrates a wide area scenario incorporating participating hospitals and emergency service vehicles. In the wide area scenario, each hospital ED is equipped with tracking and queue monitoring and collection systems, where data would be in turn made available at a decision support center and would serve as inputs to the ABM support system.

At present, ambulance diversion is principally based on best effort reporting and good-faith operation based on regional guidelines, an example of which can be found at

[25]. In other situations, diversions are posted, effectively preventing a patient from being brought to the posting facility. If these practices are done in an ad hoc or mutually exclusive fashion, they are unlikely to be optimal. In addition to these valuable heuristics, ED modeling can benefit from algorithms associated with conceptually similar areas such as the Internet and congestion avoidance schemes that deal with overcrowding of routers. In this application, the Random Early Detection (RED) algorithm [26] was adapted as a candidate for consideration when attempting to optimize ambulance or patient redirection, based on ED congestion information. This is an ideal initial algorithm for adaptation, as it has many of the attributes well suited to improving system throughput. RED accommodates limited bursts and can be effective, even when there is limited sharing of information between EDs. The RED patient diversion policy will serve as a baseline for comparison for machine learnt policy or optimizations discussed in a later section.

Furthermore, modeling extensions that add intelligence include the ability to notify and receive information from ambulances and other emergency vehicles. The actual communication services would most likely be over GSM or similar communication infrastructures where they exist. In the ABM, these services can be modeled as messages between agents, and the ABM platform would assist in optimizing ambulance diversion policies. Other considerations include estimates of emergency vehicle travel time, as these factors would be of significance in an effective model. Although beyond the current scope, the proliferation of GPS and mapping technologies allows these estimates to become empirical inputs to the multiple ED simulation. Future extensions will focus

on data sources on vehicular congestion and congestion avoidance, as additional empirical refinements to the modeling efforts [27].

5.1.8 ABM Simulation of Collaborating ED Data Infrastructure

In addition to simulations relative to staff allocation in an ED, the work further simulated a collaborating ED data infrastructure as described earlier. Real-time ED status data collected with RFID technologies would be disseminated and utilized in real time, informing ambulance and other emergency services, as well as individual citizens (perhaps through a web portal), of the near real-time status of EDs on a community-wide scale. This would allow patients and care providers to make more well-informed decisions on which ED to visit, based on current and projected wait-times. In order to forecast future patient wait times, the simulation can be run into the future a number of times, keeping track of the wait times experienced by patients arriving at future times – until some reasonable level of confidence is reached. During this process, the visualization can be disabled in order to speed multiple trials.

Since these types of systems are not yet in place, the well-vetted RED algorithm was used to model this process. As a method of network congestion management, senders of data over the network (typically the Internet) are implicitly notified of network congestion by having their data packets (discrete chunks of data) probabilistically dropped from network queues. To avoid oscillation between intense bursts of traffic and choking off traffic entirely, the rate at which these packets are dropped is ramped up gently after a certain threshold in the queue length is reached. Similarly, in the ED model, a minimum threshold was set, below which ED queue lengths are considered

acceptable and no dropping occurs. The rate at which patients are dropped increases linearly with queue length, until some maximum threshold is reached, past which the drop rate remains constant. Since "dropping" (turning away patients) was considered to be unacceptable, the model instead considers a drop as a patient redirection to another ED. The mechanism for this is either self-redirection to another ED or redirection by a central dispatch/control.

Two modes were considered: first, where patients are redirected to a random ED with uniform probability, and second, where patients are probabilistically redirected to an ED based on the ratio of doctors to patients waiting. The latter case results in EDs that are less busy assigned a higher likelihood of patients being redirected there. This reflects an assumed patient preference for shorter waiting time, and also demonstrates the utility of city-wide ED status information dissemination. This is contrasted with the former case, where patients are simply guessing as to which ED is a more preferable alternative without external guidance.

To ground the simulations to the greatest extent possible in real data, the work drew on a report on ED usage in Winnipeg, Canada released by the Manitoba Centre for Health Policy [28]. There were 185,659 ED visits in Winnipeg among six hospitals, the breakdown of which by CTAS [29] triage level roughly corresponded to the triage levels used in [21]. While the data on actual treatment times are not readily available to date, the distribution of treatment times were based on triage levels from [21]. Further, no data on the variation in patient arrival rates based on time of day, day of the week, time of year, or variation between individual EDs is available to date, and thus uniform rates are

assumed for these variables. However, it should be noted that the simulator readily incorporates these variations and ranges in data, when they become available.

With the information presented, it was possible to estimate arrival rates of patients for each triage level at each simulated ED. An arbitrary but reasonable minimum threshold of 10 patients waiting in the queue was chosen for the RED model. The drop or redirection rate increases linearly to a maximum of 50%, reached at a queue length of 20. It is reasonable to assume that staffing levels at each ED do not match demand. Because arrival rates are uniform among the simulated EDs and to make the simulation interesting, two EDs are staffed with two doctors, two EDs have three doctors, and two EDs have four doctors on staff during the simulation.

As in the staff allocation simulation, three 24-hour scenarios were investigated with ten trials each, and a warm up time of 24 hours. For comparison, the first scenario (labeled No Redirection) assumes that there is no ED status information available and that patients are better off going to the nearest ED and remaining there regardless of queue length and wait times. The second scenario allows redirection based on the RED model, and the destination ED is chosen from a uniform probability. This scenario is labeled Random RED. The third scenario, labeled Guided RED, invokes RED redirection where EDs with lower expected waiting time are probabilistically chosen more often as the destination ED.

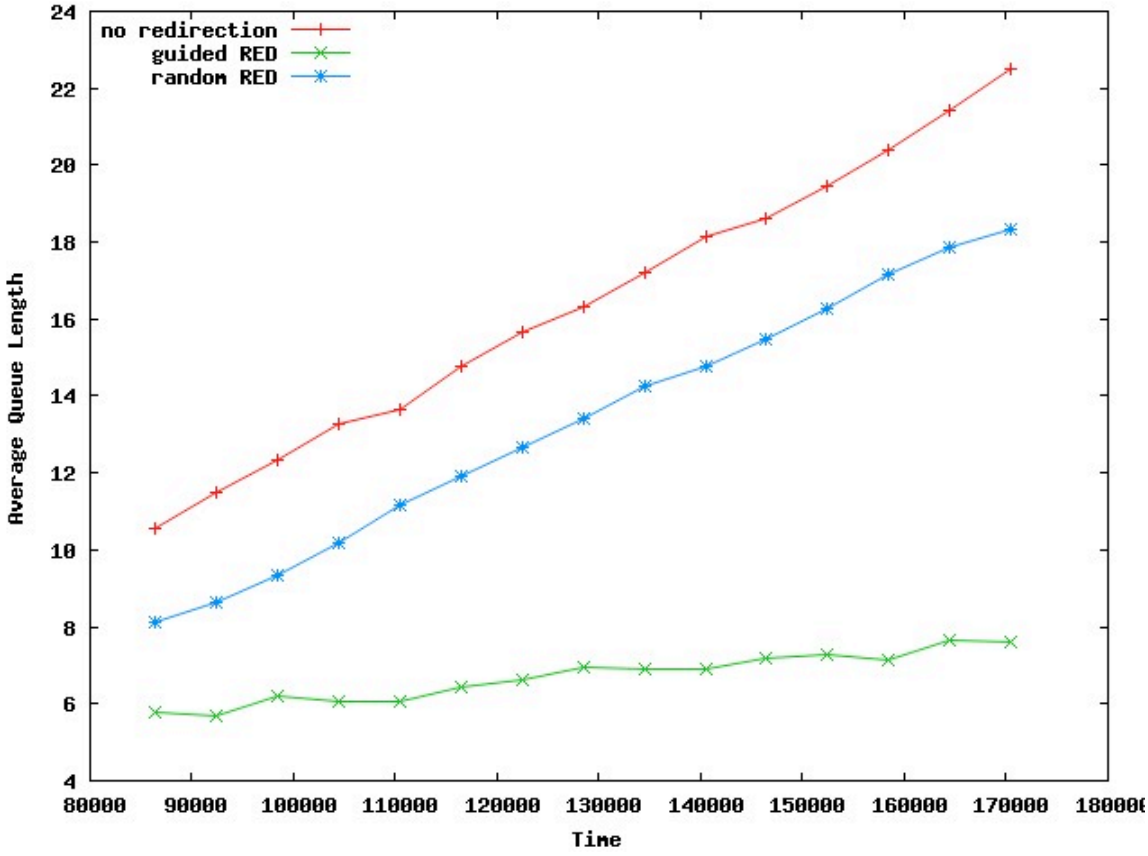

Figure 5.8. Average Queue Lengths for Various Patient Redirection Policies

Un-aggregated patient wait times for these scenarios are not shown here, as the variation between individual wait times was very high, likely due to the disparity between ED conditions. As before, results in Figure 5.8 have a qualitative value, indicating that average queue lengths among all hospitals are shortest for the Guided RED scenario. Also, the overall doctor utilization is highest in the Guided RED scenario. While the model is not currently refined enough to test causal relationships, these two factors appear to be correlated, and it is interesting to note that a significant queue length

reduction (waiting time) was achieved with only a modest increase in utilization and no additional staffing resources.

5.1.9 Detailed Emergency Department Queuing Models[1]

The following example is a QM of an individual ED, with potential extensions to a multiple-hospital QM. The example demonstrates the ability of a QM to generate quantitative data that can be used to identify system bottlenecks. While the data are quantitative, the results of this given example should be viewed as qualitative, highlighting the overall relative sensitivities and impacts of changing parameter values. As the model is refined and extended with data of higher accuracy, range, and precision, the results become amenable to statistical analysis for hard metrics, as well as causal and correlational relationships.

In the current example, a four-node system illustrated in Figure 5.9 was considered. Node 1 – Doctors, Node 2 – Diagnostic 1, Node 3 – Diagnostic 2, and Node 4 – Admission to facility. Every patient enters the facility and is classified into one of three groups: Class 1 – high priority, Class 2 – next priority and Class 3 – lowest priority. Let s1(n) and s2(n) be the first and second moments of service times at node n for all patients. These parameters represent the average service time and the variance associated with the service. For this illustration, the service times at all nodes are equal for all classes, although there are differences in the order of priority in which patients are attended, as well as differences in how patients move between nodes.

---

[1] The queuing model discussion and simulation were written by. A. Alfa and are included here for completeness

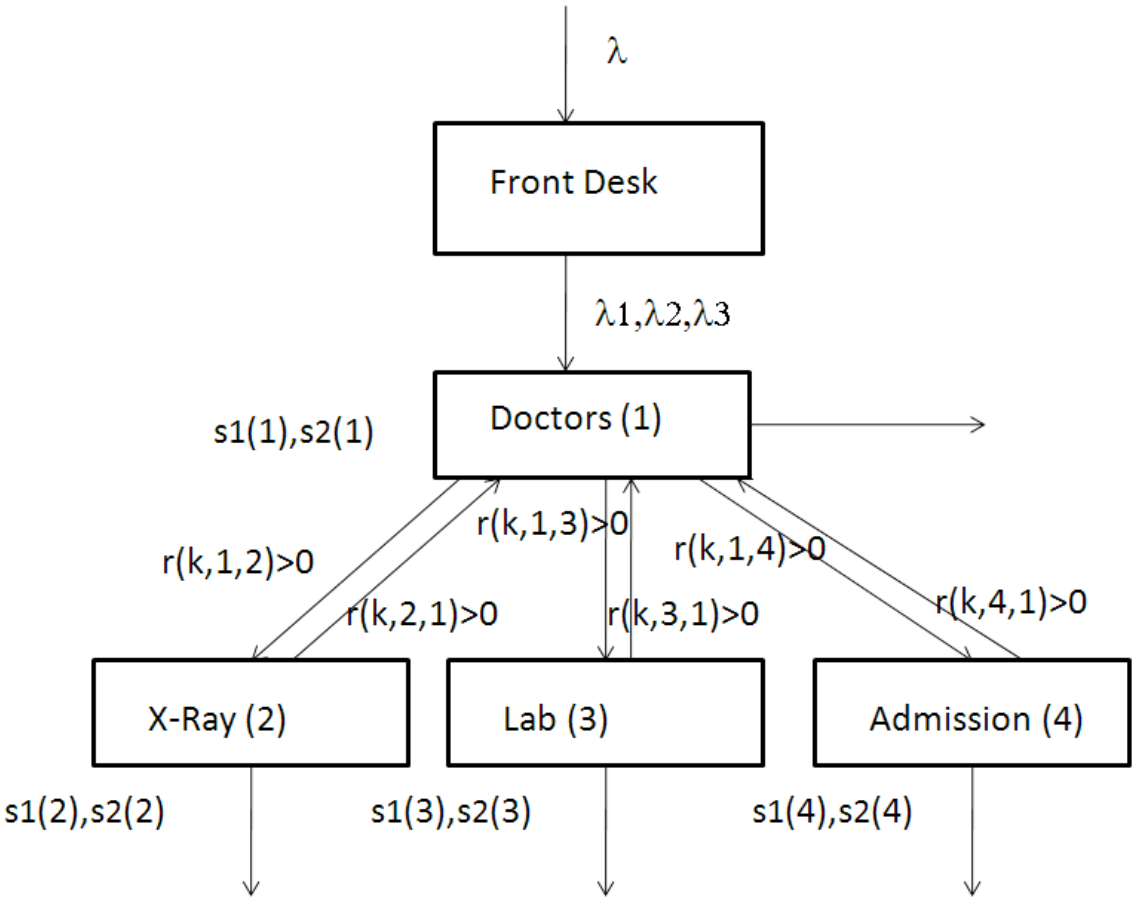

Figure 5.9. A Four Node Emergency Department Queuing Model

In the example, all patients are assumed to be seen by a healthcare worker able to assess and prescribe treatment specific to the condition (physician, physician-assistant, etc.). At that point, some patients may be discharged, while others are sent to diagnostic services and/or facility admission. Upon completing diagnostics, some patients may again be seen by a physician before discharge. In the QM, let r(k,i,j) be the ratio of class k patients that go from node i, to node j. For example, r(1,1,2) = 0.1 implies that after a

class 1 patient finishes seeing the doctor there is 0.1 probability that he/she may be sent to X-Ray.

The example used the following simulated data: patients arrived at the rate of two per hour; 50% of patients are class 1, 25% are class 2, and 25% are class 3. So the arrival rates are 1,0.5,0.5, respectively for the three classes. In Figure 5.9 these are represented by λ, $\lambda 1$, $\lambda 2$, and $\lambda 3$. Service time data were simulated as follows (in minutes): s1(1) = 20, s1(2) = 5, s1(3) = 10, s1(4) = 60, s2(1) = 500, s2(2) = 30, s2(3) = 200, s2(4) = 8000. High second moments were intentionally selected for admission to allow for high variance. As patients move through the system they are assigned transfer rates. The transfer rates, i.e. probabilities r(k,i,j) are given as: r(k,1,2) = 0.1 for all k, r(k,1,3) = 0.2 for all k, r(1,1,4) = 0.3, r(2,1,4) = 0.2, r(3,1,4) = 0.1; r(1,2,1) = 0.6, r(2,2,1) = 0.5, r(3,2,1) = 0.4; r(1,3,1) = 0.7, r(2,3,1) = 0.2, r(3,3,1) = 0.2; All other r(k,i,j) = 0. These transfer rates enable the modeling of patient flow within the ED.

For this example, the time spent at a node (waiting and receiving attention) was obtained as w(k,n) for class person of Class k at Node n. For the simulated data outlined, the results are shown in Table 5.1.

Table 5.1. Time spent at nodes as patients of various class flow through the system

	n=1	n=2	n=3	n=4
k=1	27.55	5.03	10.39	94.19
k=2	57.10	5.09	11.00	171.97
k=3	121.54	5.13	11.42	228.16

While hypothetical, the numbers illustrate the qualitative differences in system behaviour, as experienced by patients of different priority classes. In the current stage of development, the model provides insight into the relative sensitivities and impacts of varying input parameters, and allows a qualitative feel for varying policies and practices within the ED. Further simulations were carried out to demonstrate this potential. Table 5.2 illustrates the waiting and service time variations for preemptive and non-preemptive policies.

Table 5.2. Time spent at nodes for preemptive (non-preemptive) for arrivals

	n=1	n=2	n=3	n=4
k=1	27.55 (34.67)	5.03 (5.06)	10.39 (10.76)	94.19 (110.33)
k=2	57.10 (52.97)	5.09 (5.06)	11.00 (10.80)	171.97 (150.84)
k=3	121.54 (96.59)	5.13 (5.06)	11.42 (10.84)	228.16 (179.86)

Table 5.3 illustrates waiting and service times for preemptive and non-preemptive policies, as the arrivals rates are changed to (.25, 1.25, .5).

Table 5.3. Time spent at nodes, preemptive (non-preemptive) for arrivals (.25, 1.25, .5).

	n=1	n=2	n=3	n=4
k=1	21.30(29.89)	5.00 (5.05)	10.10 (10.72)	66.17 (89.52)
k=2	38.28 (41.50)	5.05 (5.05)	10.67 (10.76)	105.17 (105.57)
k=3	112.21 (88.73)	5.12 (5.06)	11.39 (10.81)	162.76 (130.13)

Table 5.4 illustrates waiting and service times and for preemptive and non-preemptive policies, as the arrivals rates are changed to (.25, .25, 1.5).

These simple simulations demonstrate the qualitative impacts in waiting and service times of patients, which would be difficult to model otherwise. While based on simulated data, the results imply that the waiting times of each patient class is affected by how they are classified. For example, by placing more patients in the highest priority class, the waiting times increase for all patients. These kinds of insights demonstrated by the QM are opportunities for further investigation.

Table 5.4. Time spent at nodes, preemptive (non-preemptive) for arrivals (.25, .25, 1.5)

	n=1	n=2	n=3	n=4
k=1	21.30(29.84)	5.01 (5.05)	10.10 (10.72)	66.17 (81.65)
k=2	25.18 (32.05)	5.03 (5.05)	10.28 (10.73)	77.24 (85.12)
k=3	62.58 (58.09)	5.08 (5.05)	10.97 (10.78)	102.35 (92.72)

As with ABMs, future efforts will refine and extend the QM model with real data, even though real data may carry a degree of uncertainty to it as well. This encompasses both topographical data, a range of patient and staff parameters, and patient flow parameters (for example, service and arrival rates). With an increasingly context-specific data set, considerable studies can be done relative to investigating efficiencies and performance improvement as a function of resources. Going further, comparative modeling of alternative care processes could be carried out as well.

The four-node QM outlined above is extendable to any level of hierarchy or any number of nodes. Furthermore, extending the QMs to encompass multiple EDs and/or alternative care practices is a reasonable extension and does not present significant technical difficulties. As with all the models discussed, the utility of the single- or multi-

hospital QM is dependent on accurate topographical and flow models with reasonable estimates for all parameters. With increased real data inputs, hard metrics become reasonable model outputs. Initial simple models that rely on simulated data nonetheless provide insight into qualitative relationships between parameters.

5.1.10 Optimizing Policy: Machine Learning

Optimizing patient access and patient care policies is not necessarily amenable to either deterministic or ad-hoc approaches, as the problems themselves are difficult combinatorial problems. In these cases, significant gains can be made with non-deterministic algorithms guided by analogues to physical systems and/or learning systems, which in turn provide a measure of credibility and confidence in the solution. This final section discusses a Genetic Programming (GP) technique that mimics evolutionary systems in attempting to optimize towards a solution. The GP approach is one of many possible approaches, but does closely relate to how actual policy and decisions are made in a very difficult problem space.

The ABM development to date for the applications outlined in earlier sections suggested a means to simulate and comparatively assess policy and practices ("what-if" scenarios) prior to implementation. However, this requires a human to generate a policy for the ABM to test. Examples of such policies may be "staff with x number of doctors instead of y number of doctors" or "begin to divert patients once the number of waiting patients exceeds a defined threshold".

In addition, ABMs can incorporate evolutionary algorithms that allow realistic agent learning, and extensions of this work include the addition of a Machine Learning (ML)

module to the ABM framework, to facilitate automatic policy generation in the context of a decision support tool for healthcare. The model generates policies, uses the ABM to evaluate them, and then uses the best individual policies as the basis for the next generation of policies. This process is iterated until pre-defined criteria are met.

Genetic programming (GP) [30] is one Machine Learning paradigm for the automatic induction of computer programs through an evolutionary process, and is one potential means of achieving automatic policy generation. The GP paradigm is well established and includes successful research applications in the areas of data mining [31], image classification [32], automatic circuit evolution [33], and robot control [34]. Evolutionary algorithms (EA), a group of algorithms to which GP belongs, can improve upon human generated policies, and sometimes in unexpected ways [35].

Future work will invoke the ABM framework to investigate the viability of using a GP-based Machine Learning system to forecast ED waiting times and to generate policy. As data collection frameworks such as the one posited in [22] become available, the ML system could be trained and validated on real data. In this instance, the ABM becomes a data generator and an input into the overall ML paradigm.

To develop this extension, refinements to the model parameters are required. One such refinement is to the model of the agent (patient), whereby the patient may change its internal state probabilistically. These internal states may represent, for example, getting less or more ill, leaving the ED, etc, and one can evolve (automatically generate) triage policies to optimize patient flow for these more complex agents. A second refinement may be to generate patient diversion policy between several EDs, where only the policy is evolved, and the means of implementing the policy is assumed to be in place and is

treated as abstract. Third, an agent class responsible for executing patient diversion policies generated by the GP-system will have to be added to the ABM.

To facilitate these extensions, the Machine Learning paradigm most closely followed is supervised or reinforcement learning (RL) [36]. Figure 5.10 illustrates the general nature of how the ABM provides feedback to the GP-based ML system, essentially acting as a data generator until real data can be used to validate, for example, a wait-time forecasting function.

Finally, a further research direction is to utilize ABMs not only for policy shifts within an existing ED, but to develop ABMs for modeling alternative forms of ED care. EDs necessarily function with competing objectives. In many EDs, the objective is to maximize the flow of patients; improvements are derived through staff levels, bed numbers and utilization, hours of operation, diversion policies etc. More recently, there have been aggressive efforts to reconceptualize the ED entirely, around its main function of addressing emergencies (vs. maximizing patient flow) [37]. This has follow-on effects in multiple directions, including but not limited to staff configurations (teams vs. individuals) and the layout of the physical facility. Here ABMs offer a useful tool in guiding decisions around such paradigmatic shifts within an individual institution.

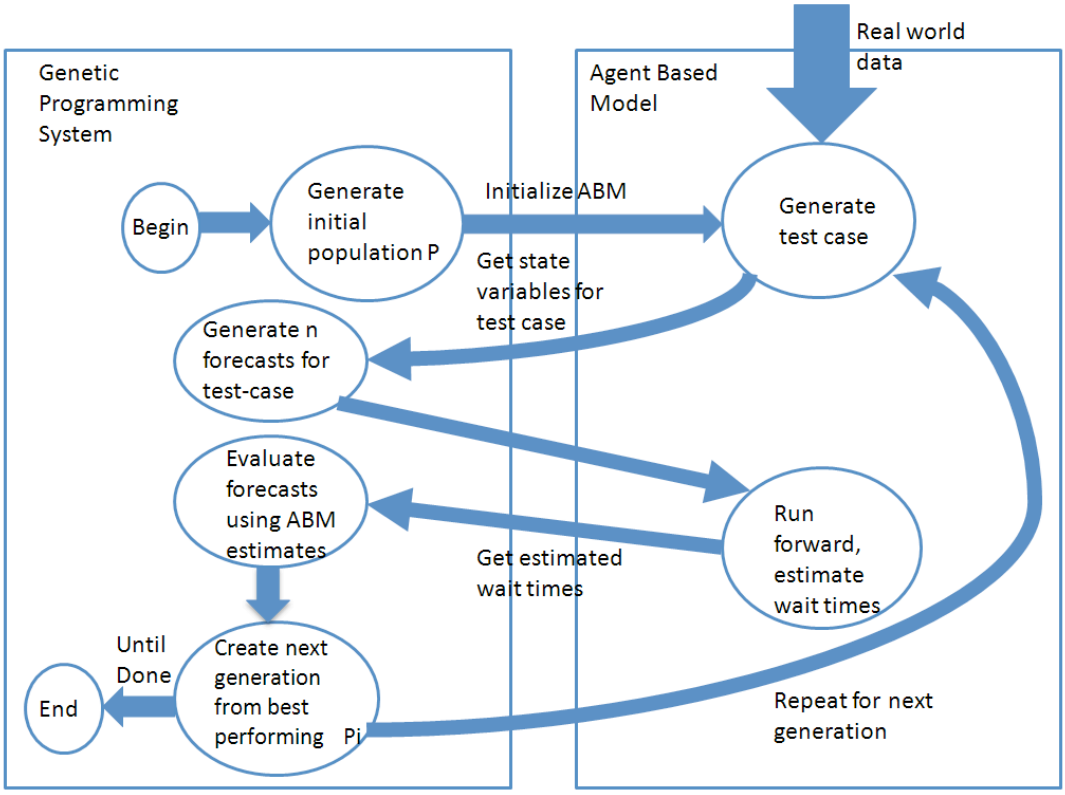

Figure 5.10. Genetic Programming and Agent Based Model Integration

5.2 Summary of Chapter 5

This chapter presented two complementary modeling methodologies applied to investigating patient access and patient waiting times in hospital EDs, within the objective of developing tools that can help guide policy and practice improvements. The first model is an Agent Based Modeling framework, oriented to the simulation of EDs in either stand-alone mode, as multiple interacting EDs, as well as technologies well suited to enhance simulation with statistical empirical data collected in real time. The second model is a more traditional queuing model, whose suitability is discussed for similar

applications. The complementary relationship between ABM and QM are demonstrated throughout this chapter, as the ABM is more suited to investigating transient phenomena, and the QM strengths are steady-state or long term analysis. Analogues from telecommunications engineering were introduced, selected for their conceptual parallels to the patient access and patient wait time cases, and because the telecommunications analogues have been well vetted within the community, albeit for different purposes.

The work is developmental, currently relying on coarse-grained approximations, relatively simple general scenarios, and to a large degree on simulated data. At their current stage of development, the models' utility lies in their ability to provide qualitative insights into the relative sensitivities and impacts of model parameters, to illuminate scenarios worthy of more complex investigation, and to iteratively validate the models as they continue to be refined and extended. With an increasing proportion of real data inputs (spatial-topographical as well as agent parameters), accurate and precise system metrics amenable to statistical processing become reasonable model outputs. Both the agent based and the queuing model frameworks are oriented to augmenting simulation with empirical data when available. Higher fidelity inputs which would be immediately useful include enhanced patient arrival rates, treatment, and service times. In this context, the work presented in this chapter further adds credibility to emerging technologies, such as RFID based patient RTLS systems, significantly enhancing the modeling efforts by provisioning the models with context-specific empirical inputs as close to real time as possible. Such systems were introduced in Chapter 4 as having a high potential as tracking data sources, These sources can be mined in a statistically

significant manner and provide real world input for the simulation. Machine Learning is one possible approach to achieving data mining, among many [38] possible approaches.

The models under development are also open source and rely on open source components. They are extendable and can be ported or tailored to a variety of hospital IT applications, several of which were identified here. Eventually, these tools may be more closely coupled to commercial hospital information systems, thereby providing better optics as to refining and optimizing hospital processes. Section 5.1.10 provides an overview of augmenting the ABM with Machine Learning, which may be the closest means of simulating decision-making in a complex problem space. The concepts introduced in section 5.1.10 provide direction for the remainder of this thesis, towards an ABM-ML based decision support tool for healthcare policy, culminating in the proof-of-concept presented in Chapter 7. To conclude, although the emergency department was the main focus of this research, the frameworks discussed are amenable to the study of intra-hospital wards, as well as inter-hospital and hospital-community interactions.

5.3 References

[1] M. Laskowski, R.D. McLeod, M.R. Friesen, B.W. Podaima, and A.S. Alfa, "Models of emergency departments for reducing patient waiting times", *PLoS ONE*, vol. 4, no. 7: e6127, 2009. [Online]. Available: doi:10.1371/journal.pone.0006127 2009.

[2] Bonabeau E (2002) Agent-based modeling: Methods and techniques for simulating human systems. Proc Natl Acad of Sci 99(Suppl 3):7280-7287. Available: http://www.pnas.org/content/99/suppl.3/7280.full#xref-ref-3-1

[3] Epstein JM (2008) Artificial society: Getting clues on how a pandemic might happen by creating a huge model of the United States. The Brookings Institution. Available: www.brookings.edu/interviews/2008/0402_agent_based_epstein.aspx.

[4] Kanagarajah AK, Lindsay, PA, Miller AM, Parker DW (2006) An exploration into the uses of agent-based modeling to improve quality of health care. Int Conf on Complex Systems, Boston, MA.

[5] Blachowicz D, Christiansen JH, Ranginani A, Simunich KL (2008) How to determine future HER ROI: Agent-based modeling and simulation offers a new alternative to traditional techniques. J Healthcare Information Management 22(1):39-45. Available: http://www.himss.org/content/files/jhim/22-1/11.pdf

[6] Spry CW, Lawley MA (2005) Evaluating hospital pharmacy staffing and work scheduling using simulation. Proc of the Winter Simulation Conference, Orlando, FL. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574514

[7] Jones SS, Evans RS (2008) An agent based simulation tool for scheduling emergency department physicians. AMIA Annu Symp Proc.:338–342.

[8] White Jr. KP (2005) A survey of data resources for simulating patient flows in healthcare delivery systems. Proc of the Winter Simulation Conference, Orlando, FL. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574341

[9] Poynton MR, Shah VM, BeLue R, Mazzotta B, Beil H et al. (n.d.) Computer terminal placement and workflow in an emergency department: An agent-based model. Available: http://www.santafe.edu/events/workshops/images/4/4a/Poynton_EtAl.pdf

[10] Emrich S, Breitenecker F, Zauner G, Popper N (2008) Simulation of influenza epidemics with a hybrid model - combining cellular automata and agent based features. 30th Int Conf on Information Technology Interfaces, Dubrovnik, Croatia: 709-714. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4588498

[11] Carley K, Fridsma D, Casman E, Altman N, Chang J et al. (2003) BioWar: Scalable multi-agent social and epidemiological simulation of bioterrorism events. Available: http://www.cos.cs.cmu.edu/publications/papers/carley_2003_biowarscalablemulti.pdf

[12] Ong BS, Chen M, Lee V, Tay JC (2008) An individual-based model of influenza in nosocomial environments. ICCS 2008, Part I, LNCS 5101:590–599.

[13] Gunal MM, Pidd M (2005) Simulation modelling for performance measurement in healthcare. Proc of the Winter Simulation Conference, Orlando, FL. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574567

[14] Borkowski M, Podaima BW, McLeod RD (in press) Epidemic modeling with discrete space scheduled walkers: Possible extensions to HIV/AIDS. BMC Public Health.

[15] McLeod RD, Laskowski M, Friesen MR, Podaima BW (2009) Agent based models for nosocomial infections: Modeling of hospital-acquired infections within a hospital. Report for the Public Health Agency of Canada, available from authors.

[16] Green LV, Soares J, Giglio JF, Green RA (2006) Using queueing theory to increase the effectiveness of emergency department provider staffing. J Acad Emergency Medicine..

[17] Green LV, Savin S (2008) Reducing delays for medical appointments: A queueing approach. Operations Research 56(6):1526-1538.

[18] Green LV (2003) How many hospital beds? Inquiry 39.

[19] Mukhi S, Laskowski M (2009) Agent-based simulation of emergency departments with patient diversion. In: Weerasinghe D, editor. Electronic Healthcare. Berlin: Springer. pp. 25-37.

[21] Patvivatsiri L (2008) A simulation model for bioterrorism preparedness in an emergency room. Available: http://www.informs-sim.org/wsc06papers/061.pdf.

[22] Sanders D, Mukhi S, Laskowski M, Khan M, Podaima BW et al. (2008) A network-enabled platform for reducing hospital emergency room waiting times using an RFID proximity location system. 19th Int. Conf. on Syst. Eng., Las Vegas, NV.

[23] Available: http://www.qe2foundation.com/en/home/howyourgifthelps/currentneeds/emergencydepartmentexpansion.aspx

[24] Available: http://standards.ieee.org/getieee802/802.11.html

[25] County of Marin, Ambulance Diversion Guidelines. Available: http://www.co.marin.ca.us/depts/HH/main/ems/documents/Policies/5400.pdf

[26] Floyd S, Jacobson V (1993) Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking 1(4):397-413.

[27] Kantowitz BH, LeBlanc DJ (n.d.) Emerging technologies for vehicle infrastructure cooperation to support emergency transportation operations. Federal Highway Administration, Contract No. DTFH61-01-C-00049 (Task 20).

[28] Doupe M, Kozyrskyj A, Soodeen R, Derksen S, Burchill C et al. An initial analysis of emergency departments and urgent care in Winnipeg. Available: http://mchp-appserv.cpe.umanitoba.ca/deliverablesList.html

[29] Canadian Triage and Acuity Scale. Available: http://www.caep.ca/template.asp?id=B795164082374289BBD9C1C2BF4B8D32

[30] Bhanzhaf W, Nordin P, Keller R, Francone F (1998) Genetic programming – An Introduction: On the automatic evolution of computer programs and its applications. San Francisco: Morgan Kaufmann Publishers, Inc.

[31] Raymer M, Punch W, Goodman W, Kuhn L (1996) Genetic programming for improved data mining: An application to the biochemistry of protein interaction. Genetic Programming 1996: Proc of the First Annu Conf. pp. 375-380.

[32] Daida J, Bersano-Begey R, Ross S, Vesecky J (1996) Computer-assisted design of image classification algorithms: Dynamic and static fitness evaluations in a scaffolded Genetic Programming environment. Genetic Programming 1996: Proc. of the First Annu Conf. pp. 279-284.

[33] Koza J, Andre D, Bennett III F, Keane M (1996) Use of automatically defined functions and architecture-altering operations in automated circuit synthesis using

Genetic Programming. Genetic Programming 1996: Proc. of the First Annu Conf. pp. 132-149.

[34] Nordin P, Banzhaf W (1997) An on-line method to evolve behavior and to control a miniature robot in real time with Genetic Programming. Adaptive Behavior 5:107-140.

[35] Laskowski M, McGrath S (2005) Effects of lying in reputation-based multi-agent systems. Canadian Conf on Elect and Comput Eng pp. 1014-1018.

[36] Sutton R, Barto A (1998). Reinforcement learning: An introduction. Cambridge, MA: MIT Press.

[37] Available: http:// www2.warwick.ac.uk/fac/med/research/hsri/emergencycare/research/edorganisation

[38] S. Weiss, N. Indurkhya, *Predictive Data Mining: A practical guide*. San Francisco: Morgan Kaufmann Publishers, Inc., 1998.

Chapter 6

A Decision Support Tool for Mitigation of the Spread of a Pathogen Causing an Influenza-Like-Illness in an Emergency Department

This chapter presents a refinement of the ABM which is suitable for comparing various "what-if" scenarios as part of a decision support tool or process for healthcare applications. This ABM version features an infection spread model, signifying an expanded focus to modelling the spread of an Influenza Like Illness (ILI) causing virus, built on top of the patient flow model developed in previous chapters. An extended example demonstrates the utility of the ABM in a use case scenario which emulates an expert or other practitioner using the ABM to perform an analysis of ILI causing virus spread within a hospital emergency department, and evaluate several mitigation policies arising from the analysis. This marks the point where "what-if" scenarios can be run in a practical and useful way without recompiling, as many aspects of the simulation can be

now configured at runtime. The heavily statistical and inferential nature of this analysis becomes apparent throughout the chapter.

After performing the aforementioned analysis, it is apparent, desirable and plausible for a machine directed search through parameter space for the optimization of some aspect of the model. Another possibility is a machine directed search for data mining, or extracting some patterns from the simulator output contrasted with the present mode of a human analyst doing the majority of the inference, with the aid of statistical software. Ultimately, the goal is to automatically generate policy in the context of the ABM-ML decision support tool for healthcare proposed in Chapter 1. To this end, this chapter describes the basic components of the Machine Learning environment, the ABM infection spread model, its statistical nature, and the need to have a machine crunch the numbers as opposed to a person. The next chapter will further extend this model to implement automated policy generation for mitigation of ILI causing virus spread in the ED for which the ABM role of evaluating "what-if" infection spread scenarios becomes essential.

The following material is an edited article submitted for publication to PLoSOne [1], and at the time of writing is under revision. This author's role in the article was the development of the original infection spread model as well as the ABM patient flow model on which it was based. This author also implemented a coarse-grained parallel version of the ABM code, to take advantage of computational parallelism on available compute clusters. This author aided in carrying out the experiments, as well as being

directly involvement in the policy development and parameterization of the four policies tested.

6.1 Agent-Based Modeling of the Spread of Influenza-Like Illness causing virus in an Emergency Department: A Simulation Study

The objective of this work was to develop an agent-based modeling framework in order to simulate the spread of influenza-like infection in a hospital emergency department. In doing so, the work complements mathematical modeling techniques for infection spread, as well as modeling applications focussed on the spread of antibiotic-resistant nosocomial infections in hospitals. Sixteen different emergency department scenarios were simulated, with further simulation of four infection control strategies. The agent-based modeling approach represents systems modeling, in which the emergency department was modeled as a collection of agents (patients and health care workers) and their individual characteristics, behaviours, and interactions. The framework was coded in C++ using Qt4 libraries running under the Linux operating system. A simple ordinary least squares regression was used to analyze the data, in which the number of patients that became infected in one day within the simulation was the dependent variable. The results suggest that the number and health status of healthcare workers play a much larger role in infection spread in an emergency department than patient characteristics. As such,

infection control strategies that target the healthcare worker appear to have greater benefits than patient-oriented policies. The agent-based modeling framework is a decision support tool available to healthcare practitioners and policymakers to assess the relative impact of infection control strategies. The framework illuminates scenarios worthy of further investigation, as well as counter-intuitive findings.

6.1.1 Introduction

This work presents an agent-based modeling (ABM) framework developed as a novel technique for modeling the spread of ILI causing virus in a hospital emergency department (ED) and summarizes the results of extensive simulation using the framework. The ABM approach is complementary to and can validate mathematical modeling methods, which have a longer history in healthcare modeling. In its focus on the spread of ILI causing virus, the work draws on and complements the literature on mathematical modeling of nosocomial infections within hospitals, and the community-level modeling of pandemic ILIs by ABM or mathematical methods. These bodies of literature surrounding the current work are reviewed briefly below.

Nosocomial, or hospital-acquired infections are prevalent in hospitals throughout the world, and contribute significantly to patient mortality and the costs of patient care [2][3]. Accordingly, a significant body of research exists to examine transmission vectors, the dynamics of infection spread, and the efficacy of infection mitigation and control measures. Nosocomial infections are generally correlated with, but not synonymous to

antibiotic resistant organisms, and therefore the literature that addresses nosocomial infections is strongly intertwined with the literature on efforts to control drug-resistant organisms.

Studies that examine infection mitigation and control factors tend to focus on facility hygiene, personal hygiene (hand hygiene, personal protective equipment for healthcare workers (HCWs)), patient isolation, and patient-to-HCW ratios [4][5]. Other studies have examined the issue more holistically, focusing on the community reservoir of organisms [6], timing of an antibiotic intervention [7], and the relative balance of infection by cross-transmission and the selective pressure of antibiotics [8]. The interests of infection control are also moving beyond nosocomial infections to encompass the concerns of recent experiences with SARS and pandemic influenza [9].

Mathematical models have been applied to examine the effects of infection control measures, as a way to gain insights into the relative impacts of mitigation and control strategies without the logistics, expense, and ethical implications of full-scale trials. Beyond specific interventions, mathematical models have also been used to examine the transmission dynamics of infection spread, the economics of infection control, and community-level and longitudinal factors in nosocomial infection spread [10][11][12].

In contrast to mathematical models of infection spread, the current study used an agent-based framework to examine the spread of ILI causing virus within an ED. ABM is systems modeling, approached from the ground up or from the perspective of its constituent parts, in order to build an aggregate picture of the whole. Systems are modeled as a collection of agents (people) and their individual characteristics,

behaviours, and interactions. In the most general context, agents are autonomous decision-making entities able to assess their situation, make decisions, and compete with one another on the basis of a set of rules. ABM's conceptual depth is derived from its ability to model emergent behaviour that may be counterintuitive or, at minimum, its ability to discern a complex behavioural whole that is greater than the sum of its parts. ABM provides a natural description of a system that can be calibrated and validated by representative expert agents, and is flexible enough to be tuned to high degrees of sensitivity in agent behaviours and interactions. ABMs are particularly well suited to system modeling in which agent behaviour is complex, non-linear, stochastic, and may exhibit memory or path-dependence [13][14][15]. Early application areas of ABM include logistics, economics, and transportation systems.

A more recent and considerable area of application for ABMs has been community-level infection spread modeling in human populations [16][17]. The focus generally constitutes community-level pandemic ILI, as this is an important public health and policy issue with far-ranging health and economic impacts. The current work differs in that the focus on ILI causing virus spread is applied at the institutional level, rather than the community level.

Within healthcare settings, ABMs have been applied alone or in complement to other techniques to the operations of EDs. In general, the literature in this area addresses system-level performance metrics, quantified in terms of patient safety [18], economic indicators [18][19], staff workload and scheduling [20][21], and patient flows [22][23]. While this literature addresses periods of typical operation, there is also a literature on

modeling of healthcare operations during critical incidents like infectious disease outbreaks and terrorist attacks [24][25][26]. However, authors agree that relatively little work exists in applying ABMs to healthcare policy development [27], for which the current work is a potential tool.

Although ABM technology is relatively mature, applying an ABM framework to model infection spread in healthcare facilities is in its infancy, and this work addresses this gap. Further, many of the ABMs developed for infection modeling are known as individual based models, in which agents are limited – by definition – to a person within the simulation [28][29]. In contrast, this ABM framework potentially includes inanimate objects as potential transmission vectors for ILIs. The significance of inanimate objects in infection transmission is examined further in [30][31].

Figure 6.1 contextualizes the ABM approach within various modeling approaches applied to healthcare modeling interests. The combination of applying an *agent-based modeling* framework to the *spread of ILI causing virus* in an *emergency department* is not otherwise found in the literature, and this constitutes a unique and novel contribution of this work. ILI causing virus was chosen as representative of infections spread by close and casual contact and as a key pandemic preparedness concern, such as hospitals' ability to cope with patient surges and compromised staffing during the 2009-2010 H1N1 pandemic. Because the ABM is highly tunable in terms of the disease epidemiology and the modes of transmission (transmission rates, etc.), the current focus on ILI is easily tailored to specific ILI causing strains and easily extended to more traditional nosocomial infections.

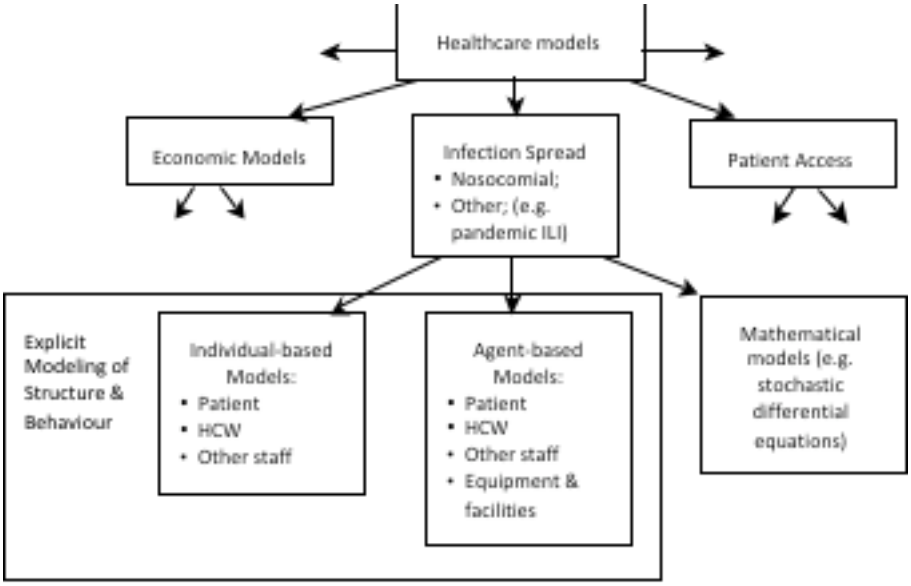

Figure 6.1. The context of ABMs within healthcare models

The objective of this work was to develop an ABM framework and to use the framework to simulate the spread of ILI causing virus in an ED. Numerous combinations of input parameters were examined relative to their impact on the number of patients that become infected with ILI causing virus during their visit to the ED. Subsequently infection control strategies were simulated in order to gain qualitative insights into the relative effects of interventions. Ultimately, the ABM framework is intended to serve as a decision support tool for infection control practice and policy development.

6.1.2 ABM Framework Implementation

The ABM framework was extended from a preliminary framework reported in [32]. Upon completion of some code re-factoring and documentation, the authors intend to

make the code available on SourceForge.net. For those interested in saving development costs, other open source and commercially available ABM frameworks/toolkits such as SWARM™, Repast™, and Anylogic™ come with varying out-of-the-box functionalities. For example, it is not immediately obvious how difficult it would be to port scripts from Anylogic to a cluster, grid, or gpu. Developing an ABM within an object oriented framework from the ground up gives the developer an additional degree of understanding of the ABM technique, in contrast to using a more commercial, or proprietary platform where hooks to underlying code may not be self evident or non-existent.

The model is not general purpose, but is a spatially directed ABM aimed at institution-level simulation of a healthcare environment. In this case, the model reflects a generic layout of an ED. Work is ongoing to customize the ABM to the specific floorplans of specific facilities, beginning with the Health Sciences Centre ED in Winnipeg, Canada. Overall tends and tendencies illuminated by the ABM are however likely to be a reasonable facsimile of the physical system and its dynamics. The application-specific ABM allows for greater flexibility in design choices, and is not protracted with rarely used or obscure features of a more general framework. However, some design choices are made already, consistent with the needs of an institutional model, thus reducing the development time for new users.

Another difference between this and other modeling approaches is the lack of a scheduler which drives the simulation. Agents spend most of their time reacting to one another, that the scheduler would effectively be activating each agent at every time step. For example, on any particular tick or time step, a patient can assess the number of

patients waiting ahead of them, and decide whether or not to leave the ED without treatment, based on their condition and a model of how long they will have to wait. A scheduler or script may be of benefit to the efficiency of this custom ABM, especially when extending the model to simulate more complex time-of-day variation of patient arrivals and staffing schedules. Partitioning the space of the World can also be used to reduce the number of agents that each agent has to interact with at each time step, speeding the simulation.

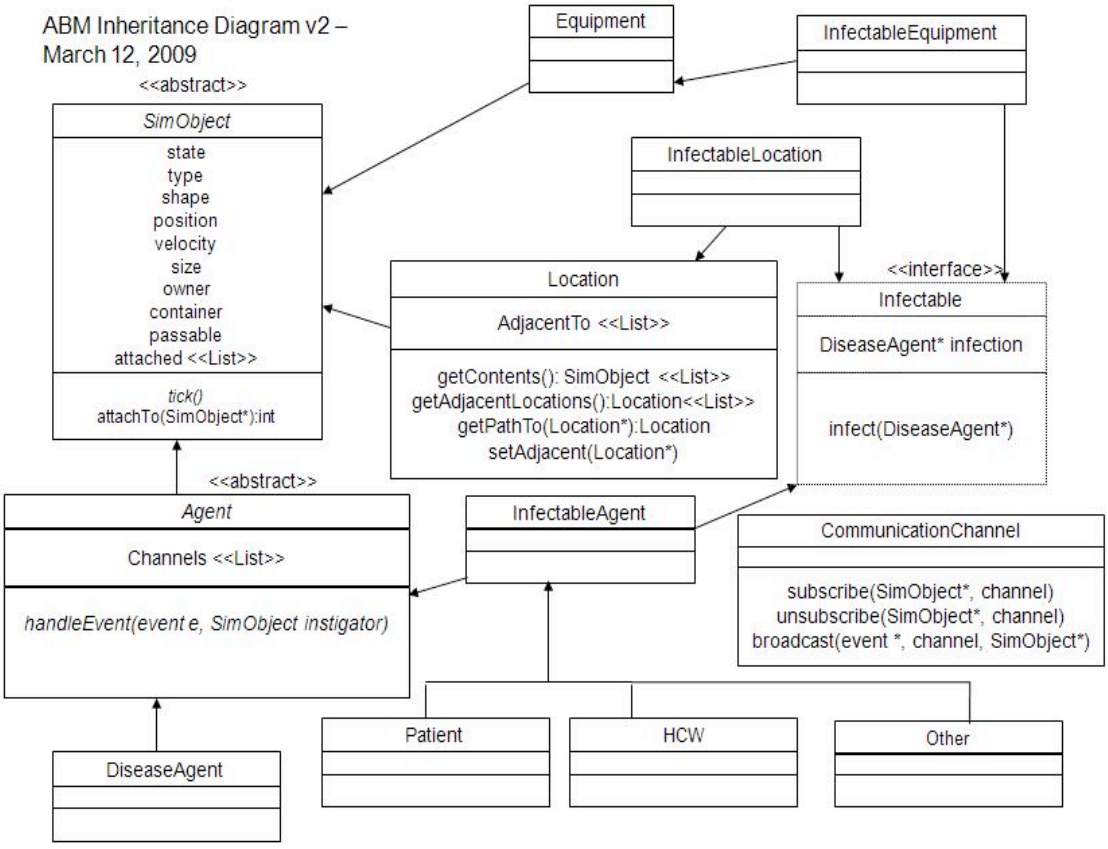

Figure 6.2. ABM inheritance diagram

For clarity, Figure 6.2 illustrates a simplified object inheritance model similar to that used in this study. Some details are omitted for intelligibility. In this framework, agents include patients, HCWs (classified as either a doctor or a nurse), and inanimate objects that can act as a transmission vector such as chairs. Each agent acts (has its tick method called) in a set, predictable, but arbitrary order. That is, agents and all other objects are simulated in the order in which they were added to the simulation, governed by the stochastic arrival rates of patients of various triage scores. Time is discrete with each time step having a resolution of 1 second (1 time step or tick = 1 second).

Patients are modeled as occupying a circular space with a radius of 30 cm, representative of their physical person as well as a concept of personal space. Close contact is defined as any interaction where the center-points of the agents are within 20 cm of one another, and casual contact as any interaction where agents are between 60 and 20 cm away from one another. These choices were arbitrary but not unreasonable, and are easily adjusted. Contact between HCW and patients will be of the close type, for the triage and treatment processes, and casual for the registration process. When a doctor is treating a patient this is considered close contact. Contact between patients will typically be of the casual variety with the exception of particularly crowded waiting rooms. In this iteration of the model, HCWs only come into contact when two doctor agents pass each other in the hallway between treatment rooms. Typically, this contact will be casual. Differentiation between contact types becomes relevant in the discussion of various ILI causing virus spread probabilities in the following section. Figure 6.3 illustrates several configurations of agent interaction.

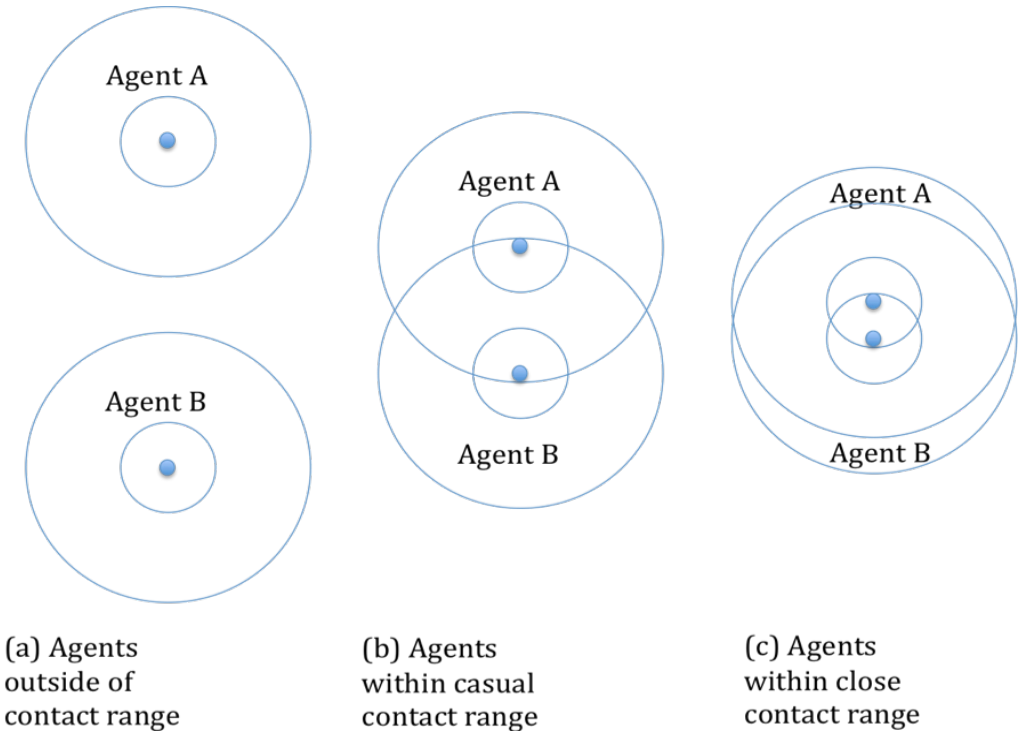

Figure 6.3. Agents in various contact scenarios:

(a) No contact. (b) Casual contact range. (c) Close contact range.

Agents practice rudimentary path planning, typically choosing the shortest path to the destination location (e.g. travel from waiting room to treatment room). The agents also selectively practice microsocial distancing, where they will tend to distance themselves according to their local density, for example when selecting a chair in the waiting room. These are examples of localized agent decision making. Figure 6.4 illustrates that the conceptual patient flow is basically the same as has been used in previous chapters. For further details about the basic patient flow model used, please see [32].

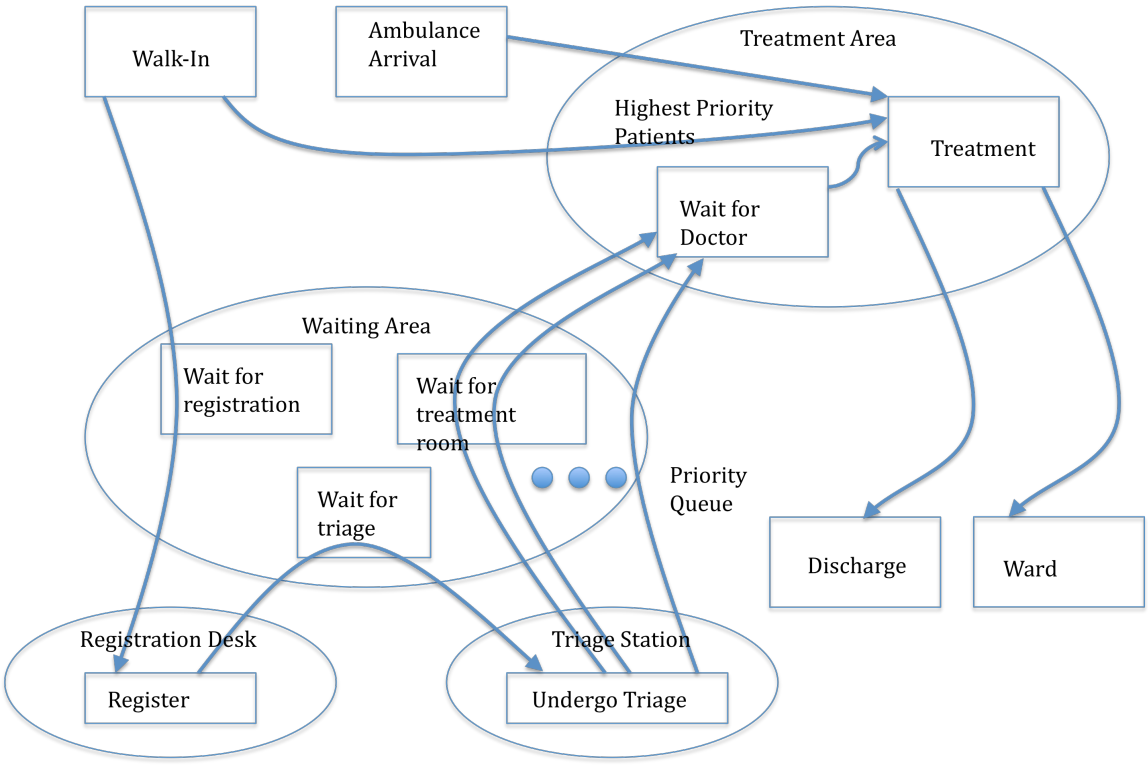

Figure 6.4. Conceptual flow of patients through the ED

Throughout the patient flow described in Figure 6.4, agents come into contact with one another. During these interactions, infection spread is modeled, primarily driven by agent distance during contact and the duration of the contact. The probability of transmission based on agent distance per time step can be found in Table 6.1. The basic infection spread model function is described in Figure 6.5.

```
Let CLOSE_RATE := ILI Close Transmission Rate Per Second
Let CASUAL_RATE := ILI Casual Transmission Rate Per Second
For each Infectious Agent A in World:
   For each Uninfected Agent B within Casual Range of A:
      Generate random number R between 0 and 1
      If B is within Close Range of A:
         If R < CLOSE_RATE: A infects B
      Else:
         If R < CASUAL_RATE: A infects B
```

Figure 6.5. Pseudocode for ILI causing virus infection spread

Agents that enter the simulation infected with an ILI causing virus are modeled as infectious upon entering (ILI presenting). Agents that acquire the ILI causing virus while visiting the ED are infectious after three hours have passed. An important aspect of this model is agent immunity. An agent can be modeled as immune to the ILI causing virus, in which case they cannot acquire the infection and accordingly cannot spread the infection. Although chairs have been mentioned as a potential infection spread vector, at the time of this writing it is an experimental feature and not modeled here.

Since ABM is very compute intensive, parallelism should be used wherever possible. Parallelism was implemented during the collection of data, as the ABM can be tailor-made to not only exploit fine-grained parallelism, but also exploitable at the coarse-grain level (process). The latter parallelism was exploited here where individual ABMs were run on a computing cluster utilizing 160 CPUs. Since the number of agents concurrently simulated is relatively small, as is the size of the simulated ED, the memory requirements were under 100MB for each instance of the simulator.

6.1.3 ABM Simulation Parameters

The emergency department layout is drawn on an 824 by 1024 pixel grid, corresponding to an area of approximately 82.4 x 102.4 meters. While the scaling is not precise, it is chosen to be a reasonable representation. The basic layout comprises a registration and triage area, a waiting room, and multiple treatment rooms accessed by corridors.

Table 6.1 summarizes the parameters that were varied, with a baseline or nominal case shown in parentheses. From the baseline or nominal case, one parameter was varied at a time, resulting in 16 individual scenarios. Throughout this section, parameter choices were developed using references from literature [33][34], conversations with Healthcare professionals [35], and data provided by anonymized Emergency Department Information System (EDIS) data from the Winnipeg Regional Health Authority [36] .

Table 6.1. Simulation parameters and values.

Parameter	**Parameter values (baseline value)**
Waiting room capacity	25, (50), 75
Number of triage nurses	(1), 2
Number of registration nurses	(1), 2
Number of doctors	2, (3), 4
Low priority patient arrival rate (patients/second)	0.000520834, (0.001041667), 0.002083334
Low priority patient ILI causing virus infection rate	0.1, (0.2), 0.4
Patient ILI immunity (probability of patient's initial condition)	0.0, (0.50), 0.75
HCW ILI immunity (probability of HCW's initial condition)	0.0, (0.50), 1.0
ILI causing virus transmission rate, casual and close	(0.0001 and 0.0002), 0.0002 and 0.0004

Waiting room capacity affects the room size in which the patients wait for treatment before they are assigned a treatment room. Roughly, the waiting room capacity

corresponds to the most patients that will fit in the waiting room based on relative size. Capacities were chosen so as to provide a reasonable range of values for a given patient flow.

The number of triage nurses affects how many patients can concurrently be serviced at the triage desk. Increasing the number of nurses has the standard queuing theory implication of reducing the expected wait time of the enqueued patients. However, in this model each nurse is a potential new contagious ILI causing virus carrier. The number of registration Nurses functions as above for the registration desk and related service.

The number of doctors affects how many patients can undergo the treatment process concurrently. This is conceptually analogous to varying the number of nurses as above. The baseline was chosen to approximately be able to handle the patient flow, and representative of typical staffing in an urban ED. The number of doctors was both increased and decreased to assess the effect of this parameter.

To model an ILI patient surge, the low priority (low urgency) patient arrival rate was set higher than the arrival rates for other patient categories: 0.000208333 and 0.000041667 for medium and high priority patients respectively. These rates were informed by the literature in the area [33][34], estimates obtained from the Winnipeg Regional Health Authorities Electronic Data Information System (EDIS), and discussion with emergency room practitioners. In the baseline simulation, these rates correspond to approximately five arrivals/hour. The rational was to capture the notion that many of the patients presenting ILI tend to be triaged at a low priority level. Of course these parameters are adjustable and could be tailored to more accurate records of a specific

facility as required. Arrivals are modeled as a Poisson arrival process with a rate per second equal to the low priority arrival rate (and corresponding rates for the other patient priority categories). Simulated scenarios included doubling and halving this rate for the low priority patient arrival rate.

The low priority patient ILI causing virus infection rate dictates, for every low priority patient that enters the simulation via the Poisson process mentioned above, the probability that the arriving patient shall be infected upon arrival. This rate is initially 0.2, meaning that 20% of low priority patients will arrive already infected. This rate was both doubled and halved in the respective scenarios. Initial infection rates for the medium and high priority Patients were 0.1 and 0.05, respectively. Again these numbers were informed by the literature [29] as well as by values deemed reasonable as could be extracted by observation and conversation with practitioners. This type of data is expected to become more readily available here as well as elsewhere as advances in electronic record keeping proceeds and availability and sharing of this type of information improves [37].

Patient immunity is the same across all patient categories, and dictates the probability of any priority class of patient being immune to ILI causing virus infection upon arrival. If it is decided that the arriving patient is immune, the patient cannot therefore also begin the simulation being infected. The initial rate of 50% indicates that half of the arriving low priority patients will enter the simulation infected (and infectious). This rate is halved but only increased to 75% for the simulated scenarios. If it had been doubled to 100%, the result would have been trivial: no patients can become infected.

Similar to the patient ILI immunity, HCW ILI immunity dictates the probability that any given HCW is immune to ILI causing virus infection at the beginning of the simulation. Unlike patient ILI immunity, the HCW ILI immunity was also investigated at 100%, as this is a probable scenario, particularly in the event of epidemic or pandemic influenza. Neither patients nor HCW can acquire immunity during the course of the simulation.

The final parameter, the ILI close and casual transmission rate are the probability that on any given time step, an infectious agent will infect an uninfected agent within close and casual contact range, respectively. The precise functionality of these is specified above. The casual transmission rate is half that of the close transmission rate as suggested by [29]. The specific values used here were chosen such that some, but not all, agents became infected during a prototypical simulation.

Other simulations not presented here, included features such as providing the patient with the decision making capability of leaving untreated. Data on patients that left without treatment is also available and extracted from EDIS information. In the case of the regional health authority, hospital EDs post the priority of patients and their number as well as expected time to treatment; this has the effect of promoting attrition in the case of low priority patients enduring long anticipated waiting times. Again, this is a stochastic process providing agents with a small degree of decision making capability and autonomy. Other simulations utilize the chairs as a vector for ILI causing virus spread, however the results presented here do not.

6.1.4 Results

Based on Table 6.1, 16 scenarios were simulated approximately 500 times each, with roughly 200 patients visiting the ED each run. A time period of 48 hours was simulated, with the first 24 hours considered a 'warm-up' or calibration period. For data collection, hour 24 was considered the start of the simulation and hour 48 was considered the end of the simulation.

The variables of interest in the simulation were the ILI causing virus infection rates among HCWs and among patients as a result of their contact with one another during the simulation. Specifically, the recorded output data included patient count at 24 and 48 hours, infected patient count at 24 and 48 hours, HCW count at 24 and 48 hours, infected HCW count at 24 and 48 hours, the count of infected patients that entered the simulation between 24 and 48 hours, the count of all patients that entered the simulation between 24 and 48 hours, the count of infected patients that left the simulation between 24 and 48 hours, and the count of all patients that left the simulation between hours 24 and 48 hours. These data were used to calculate the dependent variable as the number of patients that became infected with ILI causing virus between 24 and 48 hours in the simulation.

The data was stratified into two separate, non-overlapping samples, the first including the simulations in which the baseline parameters were changed ("baseline changes"); the second the simulations of four infection mitigation policies and the baseline values for the four policies ("four policies"). The first analysis (using only the data with the baseline changes) helped identify variables that were effective in reducing the number of

patients that became infected in the emergency department. From these baseline results, four policies were identified and formulated for further simulation, in terms of their relative effectiveness in reducing the number of patients that became infected vis-à-vis implementing no policy. The four policies and their formulation are detailed later in this section.

A simple ordinary least squares (OLS) regression[1] was used to analyze both sets of data, in each of which the number of patients that became infected between 24 and 48 hours was the dependent variable. Independent variables in the "baseline changes" analysis included all parameters initially set at their baseline value (see Table 1), then varied, one by one, to measure the effect of those changes. Other independent variables included HCW and patient counts at 24 and 48 hours. All independent variables, as well as their means and ranges, are shown in Table 6.2. From Table 6.2, a high degree of variability is evident in the range of variables, highlighting the ABM approach with its inherent focus on transient phenomena.

[1] Normality of the residuals was tested using the Jarque-Bera test statistic and was rejected for both data sets (JB = 2,843.125 for "baseline changes"; JB = 2,030.428 for "four policies"). Since a transformation of the independent variables that would make the residuals normal could not be found, bootstrapping was applied to all regressions. This resulted in only one change in level of significance (from 1% to 5%) and is noted in the text.

Table 6.2. Summary statistics of variables in each model.

	Baseline changes			Four policies		
Variable	**Mean**	**Min**	**Max**	**Mean**	**Min**	**Max**
Number of patients that become infected between 24 and 48 hours	17.09	0	111	6.45	0	46
Infected HCW count at 24 hours	1.51	0	6	1.87	0	11
Infected HCW count at 48 hours	2.06	0	6	2.75	0	19
Number of patients entering between 24 and 48 hours	114.51	44	245	111.71	80	151
Patient count at 24 hours	22.78	0	189	15.28	0	57
Patient immunity	0.48	0	0.75	0.5	0.5	0.5
HCW immunity	0.50	0	1	0.5	0.5	0.5
Low priority infection rate	0.21	0.10	0.40	0.2	0.2	0.2
Low priority arrival rate	0.0011	0.0005	0.002	0.001	0.001	0.001
Number of doctors	3	2	4	3	3	3
Waiting room capacity	50	25	75	50	50	50
Number of registration nurses	1.29	1	2	2	2	2
Number of triage nurses	1.06	1	2	2	2	2
Casual transmission rate	0.00021	0.00005	0.0004	0.0002	0.0002	0.0002

Table 6.3. OLS regression results.

Variable	Coefficient	SE	*p*-value
Infected HCW count at 24 hours‡	2.251	0.122	0.000
Infected HCW count at 48 hours‡	1.277	0.116	0.000
Number of patients entering between 24 and 48 hours‡	0.223	0.007	0.000
Patient count at 24 hours‡	0.527	0.007	0.000
Patient immunity‡	-2.738[1]	0.059	0.000
HCW immunity‡	-0.193[1]	0.055	0.000
Low priority infection rate‡	-0.848[1]	0.146	0.000
Low priority arrival rate	0.790[1]	0.852	0.354
Number of doctors	-2.925	0.270	0.000
Waiting room capacity	0.001	0.009	0.900
Number of registration nurses	-2.979	0.336	0.000
Number of triage nurses	-2.780	0.337	0.000
Casual transmission rate‡	1.919[1]	0.017	0.000
Constant	-0.658	1.225	0.591
R squared	0.883		
Number of observations	8,013		

‡Significant at the 1% level; †significant at the 5% level; *significant at the 10% level

[1]Patient immunity, HCW immunity and low priority infection rate coefficients and SE are scaled by a factor of 10; the low priority arrival rate coefficient by a factor of 1,000; and the casual transmission rate coefficient by a factor of 10,000, for easier interpretation.

Results for the OLS regression comparing changes in the baseline values are shown in Table 6.3. The R2 for the model is 0.883, which suggests a good fit. The results show

the effect that a one-unit (or smaller for the scaled coefficients) change in each of the independent variables has on the number of patients that became infected between 24 and 48 hours in the emergency department.

HCW in the ED have a significant and relatively substantial effect on the number of patients that get infected. One-unit increases in the number of doctors, registration nurses and triage nurses decreases the number of patients that get infected by 2.925, 2.979 and 2.780, respectively. However, every additional infected HCW at 24 hours increases the number of patients that get infected by 2.251, and every additional infected HCW at 48 hours increases the number of patients that get infected by 1.277. The number of patients in the emergency department has a much smaller effect: for every additional patient already in the ED at 24 hours, and for every additional patient that enters between 24 and 48 hours, the number of patients that get infected increases by 0.527 and 0.223, respectively. However, in terms of immunity, patients play a much larger role than HCW. A 0.1 unit increase in patient immunity reduces the number of patients that get infected by 2.738, while an equal increase in HCW immunity reduces that number by only 0.193. An increase in the casual transmission rate by 0.0001 increases the number of patients that get infected by 1.919. Finally, a 0.1 unit increase in the low priority infection rate reduces the number of patients that get infected by 0.848, effecting a decrease rather than an increase because the more patients arrive already infected, the fewer patients are left to get infected. Variables that do not significantly affect the number of patients that get infected in the emergency department between 24 and 48 hours include the low priority arrival rate and waiting room capacity.

Based on the initial results, four cases were identified for further simulations. These cases modeled various infection control practices that could be implemented in an ED, in the interest of determining whether any of the interventions led to statistically significant outcomes of the dependent variable. Each case was simulated independently of the other three cases.

First, because the low priority patient ILI causing virus infection rate was statistically significant, the intervention modeled was the triage Nurse masking all ILI-symptomatic patients. Within the ABM, this was modeled as reductions to the ILI causing virus transmission rates (close and casual contact) for low priority agents once they reach the triage stage. If an agent is masked and infected, then the probability of the masked agent infecting another agent is scaled by 0.5 for casual contact, and scaled by 0.75 for close contact. Therefore, the masked agent's probability of infecting an agent within casual contact range (per time step) is: 0.5*0.0001 = 0.00005 and 0.75*0.0002 = 0.00015 within close contact range. Similarly, the probability of an infected, unmasked agent infecting a masked agent (per time step) is: 0.5*0.0001 = 0.00005 for casual contact, and 0.75*0.0002 = 0.00015 for close contact. An infected masked agent's probability of infecting an uninfected masked agent (per time step) is: 0.5*0.5*casual transmission rate = 0.25*0.0001 = 0.000025 if they are within casual contact range and, 0.75*0.75*close transmission rate =0.5625*0.0002 = 0.0001125 if they are within close contact range.

Second, because number of infected HCWs at both 24 and 48 hrs was statistically significant, the intervention modeled was masking all HCW. This was similarly modeled as reductions to the ILI causing virus transmission rates (close and casual contact) for

HCW agents. HCW masks are modeled identically to the patient masks mentioned previously, with the exception that HCW begin the simulation masked if the policy is in effect.

Third, because low priority patient ILI causing virus infection rate was statistically significant, the intervention modeled was streaming ILI-symptomatic patients into an alternate treatment stream or area, as may be done during an influenza epidemic. This intervention effectively cohorts these patients by removing them from the rest of the patient population. Unlike the masking policy, this policy was only applied to infected low priority patients. The triage nurse identifies these patients and then removes them from the simulation (i.e. sent home or to isolated treatment area through the regular exit routes). The model for this policy could be further elaborated by adding a probabilistic misdiagnosis model whereby infected patients are probabilistically allowed to stay in the normal treatment stream. Similarly, uninfected patients may erroneously be sent home.

Finally, because number of infected HCWs at 24 and 48 hrs was statistically significant, the intervention modeled was sending the HCW home once they became sick. Within the ABM, this was modeled as removing the HCW from the simulation three hours after they became infected and running short-staffed for 90 minutes until an additional HCW enters the simulation to replace the absent HCW. If an infected HCW is the last remaining HCW for that particular job, (ex. Triage Nurse), the HCW will wait until a replacement arrives before leaving the simulation, effectively keeping them in the simulation in an infected and infectious state for that period of time.

Table 6.4. OLS regression – four policies.

Variable	Coefficient	SE	*p*-value
Infected HCW count at 24 hours‡	1.515	0.089	0.000
Infected HCW count at 48 hours‡	0.538	0.076	0.000
Number of patients entering between 24 and 48 hours‡	0.073	0.007	0.000
Patient count at 24 hours‡	0.091	0.009	0.000
Policy: mask infected HCW‡	-2.781	0.240	0.000
Policy: dismiss infected HCW‡	-9.764	0.246	0.000
Policy: mask low priority patients‡	-2.107	0.237	0.000
Policy: dismiss low priority patients‡	-2.448	0.243	0.000
Constant‡	-3.959	0.836	0.000
R squared	0.566		
Number of observations	2,499		

‡Significant at the 1% level; †significant at the 5% level; *significant at the 10% level

Results for the OLS regression comparing the four policies and policy baseline are shown in Table 6.4. The R2 for this model is 0.566, considerably lower than for the baseline changes analysis, and suggests a moderately good fit.

All variables are highly significant (p-value < 0.001)). The magnitude of the effect of each independent variable that was also included in the baseline changes regression is much more modest in the policy comparison. A one-unit increase in infected HCW count at 24 and 48 hours increases the number of patients that get infected between 24 and 48 hours by 1.515 and 0.538, respectively. Again, the effect of patient count is much

smaller: one-unit increases in the number of patients entering between 24 and 48 hours and patient count at 24 hours increases the number of patients that get infected by only 0.073 and 0.091, respectively. Among the four policies, dismissing infected HCW has the largest effect. Implementing this policy (vis-à-vis implementing no policy) reduces the number of patients that get infected by 9.764. The effects of the other three policies are smaller, and they are similar in magnitude. Implementation of each policy vis-à-vis no policy results in reduction between 2.1 and 2.8 in the number of patients that get infected.

Overall, the results of both analyses tend to suggest that the number of HCW plays a much larger role in the spread of ILI causing virus in an emergency department than number of patients in the ED. As a result, implementing a policy to dismiss infected HCW had the greatest benefits, while patient-oriented policies and a policy to mask infected HCW had substantially smaller benefits, though still showed significant reductions in the number of patients that get infected.

6.2 Summary of Chapter 6

The work provides insights primarily in two directions. First, through the development of the ABM framework specifically designed to simulate the spread of contact centric infections such as ILI causing virus through an ED, the limitations of the model itself are clarified. The apparent novelty of applying an agent-based modeling framework to the infection spread in an emergency department is discussed as a contribution of this work. Simultaneously, the potential capabilities of the ABM technique overall are illuminated,

in that any degree of specificity in agent characteristics, behaviours, and interactions can be accommodated, constrained only by time and computing resources and ultimately creating a model of extremely high resolution and fidelity. In contrast to other techniques (mathematical modeling), Agent Based Modeling allows users to construct a comprehensive representation of the real world as the available data allows.

Further work with the current ABM framework will be directed to fine-tuning the model and augmenting with data obtained more recently from the Emergency Department Information System (EDIS) from the Winnipeg Regional Health Authority. Comprising seven facilities and approximately 120,000 ED visits over a six-month period, the data is being used to extract distributions of patient arrival rates, patient triage scores, patient age, patient lengths of stay, and patient waiting times. These distributions will be used to fine-tune the parameters of the ABM framework, in order to enhance the validity of the output.

Second, the work provides insights into the variables that impact the spread of ILI causing virus in an ED, and the concomitant impact of corresponding interventions, as discussed earlier. The research was guided by discussions with ED practitioners, ED information systems data, and available literature. For qualitative validation and verification, the ABM incorporates an animation, allowing the user to view the simulation while it is running. This animation assists in model development as well providing a decision maker with an overall view of the ED model while running. The data collected and the analyses performed allowed baseline operations to be simulated, and infection control policies to be introduced and assessed for impact. The results suggest that the health status and numbers of HCWs in an emergency department exert a

larger influence over overall infection spread, than patient-oriented policies. As such, the ABM is a decision support tool available to healthcare practitioners and policy-makers in order to qualitatively assess the relative impact of various infection control strategies, such as early and mandatory vaccination for HCWs. The ABM illuminates scenarios worthy of further investigation as well as counter-intuitive findings, such as the statistical insignificance of waiting room capacity on the number of patients that become infected. In this regard, it is worthwhile to further consider the research that aims to assess control strategies for nosocomial infections in hospitals: some researchers have concluded that chance effects and naturally occurring large fluctuations in prevalence can confound the effects of interventions [7][8], and no model is yet sensitive enough to quantitatively and exactly validate the full range of infection dynamics. Nonetheless, the ABM technique, already validated for urban- and community-level modeling, also holds significant promise for this type of institution-level investigation. Since the agent representation is constant across these various scales of modeling, ABM also offers the potential to integrate different levels of models with one another.

The system presented is amenable to Machine Learning technologies making possible the ABM-ML based decision support tool for automated generation of healthcare policy proposed in Chapter 1. As mentioned in Chapter 5, a candidate Machine Learning technology is Genetic Programming [38], with well documented performance on a variety of complex, real-world problems such as this one, and can be implemented effectively with comparatively little expert domain knowledge [39]. One use case would be data mining in order to indentify significant inputs. Another might be evolution and optimization of various ED policies, furthering ABM utility as a decision support tool for

healthcare policy management. An ABM-ML tool to evolve a policy for mitigation of ILI virus spread in the ED will be implemented in the next chapter, in which the infection spread model presented here, and its ability to evaluate "what-if" scenarios, is central to comparing automatically generated policies for the GP evolutionary process.

6.3 References

[1] M. Laskowski, B.C.P. Demianyk, J. Witt, S. Mukhi, M.R. Friesen and R.D. McLeod, "Agent-based modeling of the spread of influenza-like illness in an emergency department: A simulation study," *PlosONE*, submitted.

[2] D.E. Zoutman et al., "The state of infection surveillance and control in Canadian acute care hospitals," American J. of Infection Control, vol. 31, no. 5, pp. 266-273, 2003.

[3] D. Gravel et al., "Point prevalence survey for healthcare-associated infections within Canadian adult acute-care hospitals," J. of Hospital Infection, vol. 66, pp. 243-248, 2007.

[4] E.M.C. D'Agata, G. Webb, and M. Horn, "A mathematical model quantifying the impact of antibiotic exposure and other interventions on the endemic prevalence of vancomycin-resistant enterococci," J. Infectious Diseases, vol. 192, pp. 2004-2011, 2005.

[5] M. Lipsitch, C.T. Bergstrom, and B.R. Levin, "The epidemiology of antibiotic resistance in hospitals: Paradoxes and prescriptions," PNAS, vol. 97, no. 4, pp. 1938-1943, 2000.

[6] J.V. Robotham, C.A. Scarff, D.R. Jenkins, and G.F. Medley, "Meticillin-resistant Staphylococcus aureus (MRSA) in hospitals and the community: Model predictions based on the UK situation," J. of Hospital Infection, vol. 65(S2), pp. 93-99, 2007.

[7] B.S. Cooper, G.F. Medley, and G.M. Scott, "Preliminary analysis of the transmission dynamics of nosocomial infections: Stochastic and management effects," J. of Hospital Infection, vol. 43, pp. 131-147, 1999.

[8] I. Pelupessy, M.J.M. Bonten, and O. Diekmann, "How to assess the relative importance of different colonization routes of pathogens within hospital settings," PNAS, vol. 99, no. 8, pp. 5601-5605, 2002.

[9] D.E Zoutman and B.D. Ford, "A comparison of infection control program resources, activities, and antibiotic resistant organism rates in Canadian acute care hospitals in 1999 and 2005: Pre- and post-severe acute respiratory syndrome," American J. of Infection Control, vol. 36, no. 10, pp. 711-717, 2008.

[10] D.J. Austin and R.M. Anderson, "Studies of antibiotic resistance within the patient, hospitals and the community using simple mathematical models," Phil. Trans. R. Soc. Lond. B, vol. 354, pp. 721-738, 1999.

[11] L. Temime, G. Hejblum, M. Setbon, and A.J. Valleron, "The rising impact of mathematical modeling in epidemiology: Antibiotic resistance research as a case study," Epidemiol. Infect., vol. 136, pp. 289-298, 2007.

[12] M.J.M. Bonten, D.J. Austin, and M. Lipsitch, "Understanding the spread of antibiotic resistant pathogens in hospitals: Mathematical models as tools for control," Clinical Infect. Diseases, vol. 33, pp. 1739-1746, 2001.

[13] E. Bonabeau, (2002). Agent-based modeling: Methods and techniques for simulating human systems. Proceedings of the National Academy of Science [Online]. 99(Suppl 3), pp. 7280-7287. Available: http://www.pnas.org/content/99/suppl.3/7280.full#xref-ref-3-1

[14] J.M. Epstein, "Modelling to contain pandemics," Nature, vol. 460, pp. 687, 2009.

[15] N. Hupert, W. Xiong, and A. Mushlin, "The virtue of virtuality: The promise of agent-based epidemic modeling," Translational Research, vol. 151, no. 6, pp. 273-274, 2008.

[16] J.M. Epstein, "Artificial society: Getting clues on how a pandemic might happen by creating a huge model of the United States," The Brookings Institution. [Online]. Available: www.brookings.edu/interviews/2008/0402_agent_based_epstein.aspx.

[17] S. Merler et al., Modeling influenza pandemic in Italy: An individual-based approach. Available http://www.sis-statistica.it/files/pdf/atti/SIS%202007%20Venezia%20intermedio_121-131.pdf

[18] A.K. Kanagarajah, P.A. Lindsay, A.M. Miller, and D.W. Parker, "An exploration into the uses of agent-based modeling to improve quality of health care," in International Conference on Complex Systems, Boston, MA, 2006.

[19] D. Blachowicz, J.H. Christiansen, A. Ranginani, and K.L. Simunich, "How to determine future HER ROI: Agent-based modeling and simulation offers a new alternative to traditional techniques," J. Healthcare Information Management, vol. 22, no. 1, pp. 39-45, Winter 2008. [Online]. Available: http://www.himss.org/content/files/jhim/22-1/11.pdf

[20] C.W. Spry and M.A. Lawley, "Evaluating hospital pharmacy staffing and work scheduling using simulation," in 2005 Proceedings of the Winter Simulation Conference, Orlando, FL, 2005. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574514

[21] S. S. Jones and R S. Evans, "An agent based simulation tool for scheduling emergency department physicians," AMIA Annu. Symp. Proc. 2008; 2008, pp. 338–342.

[22] K.P. White Jr., "A survey of data resources for simulating patient flows in healthcare delivery systems," in 2005 Proceedings of the Winter Simulation Conference, Orlando, FL, 2005. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574341

[23] M.R. Poynton, V.M. Shah, R. BeLue, B. Mazzotta, H. Beil, and S. Habibullah, "Computer terminal placement and workflow in an emergency department: An agent-based model," [Online]. Available: http://www.santafe.edu/events/workshops/images/4/4a/Poynton_EtAl.pdf

[24] S. Emrich, F. Breitenecker, G. Zauner, and N. Popper, "Simulation of influenza epidemics with a hybrid model - combining cellular automata and agent based features," 30th International Conference on Information Technology Interfaces 2008, Dubrovnik, Croatia, 2008, pp. 709-714. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4588498

[25] K. Carley, D. Fridsma, E. Casman, N. Altman, J. Chang, B. Kaminsky, D. Nave, and A. Yahja, "BioWar: Scalable multi-agent social and epidemiological simulation of bioterrorism events," [Online]. Available: http://www.cos.cs.cmu.edu/publications/papers/carley_2003_biowarscalablemulti.pdf

[26] B.S. Ong, M. Chen., V. Lee, and J. C. Tay, "An individual-based model of influenza in nosocomial environments," ICCS 2008, Part I, LNCS 5101, pp. 590–599, 2008.

[27] M.M. Gunal and M. Pidd, "Simulation modelling for performance measurement in healthcare," 2005 Proceedings of the Winter Simulation Conference, Orlando, FL, 2005. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1574567

[28] B.S. Ong, M. Chen., V. Lee, and J. C. Tay, "An individual-based model of influenza in nosocomial environments," ICCS 2008, Part I, LNCS 5101, pp. 590–599, 2008.

[29] C. van den Dool, M.J.M. Bonten, E. Hak, J.C.M. Heijne, and J. Wallinga, "The effects of influenza vaccination of health care workers in nursing homes: Insights from a mathematical model," PLoS Medicine, vol. 5, no. 10, e200 doi:10.1371/journal.pmed.0050200.

[30] B. Bean, B.M. Moore, B. Sterner, L.R. Peterson, D.N. Gerding, and H.H. Balfour Jr., "Survival of influenza viruses on environmental surfaces," J. Infectious Diseases, vol. 146, no. 1, pp. 47-51, 1982.

[31] A. Kramer, I. Schwebke, and G. Kampf, "How long do nosocomial pathogens persist on inanimate surfaces? A systematic review," BM Infectious Diseases, vol. 6:130, 2006.

[32] M. Laskowski, R.D. McLeod, M.R. Friesen, B.W. Podaima, and A.S. Alfa, "Models of emergency departments for reducing patient waiting times," PloS ONE, vol. 4, no. 7, 2009.

[33] K.P. White, Jr., "A survey of data resources for simulating patient flows in healthcare delivery systems", Proceedings of the 2005 Winter Simulation Conference, pp.926-935, 2005.

[34] L. Patvivatsiri, “A Simulation Model for Bioterrorism Preparedness in an Emergency Room”, Proceedings of the 2006 Winter Simulation Conference, pp.501-508, 2006.

[35] Personal communication, March 2009, August 2009, November 2009.

[36] Raw data, Emergency Department Information System, Winnipeg Regional Health Authority, January 2009 – June 2009.

[37] H.A. Piwowar, M.J. Becich, H. Bilofsky, and R.S. Crowley, “Towards a data sharing culture: Recommendations for leadership from academic health centers,” PLoS MED, vol. 5, no. 9, 2008, doi:10.1371/journal.pmed.0050183

[38] W. Bhanzhaf, P. Nordin, R. Keller, and F. Francone, Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and its Applications. San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1998.

[39] J.R. Koza et al., Genetic Programming IV: Routine Human Competitive Intelligence. Norwell, MA: Kluwer Academic Publishers, 2003

Chapter 7

ABM-ML Hybrid for Automated Policy Generation

The previous chapters have consistently highlighted the possibility and potential utility of combining a Machine Learning module to augment an Agent Based Modelling tool for decision support in healthcare. Furthermore, the utility of specific applications that would be improved, or partially automated by leveraging Machine Learning have been introduced in previous chapters, and this chapter will develop this potential. Section 7.1 introduces, in general terms, the ML approach used throughout this chapter: Genetic Programming (GP). Section 7.2 presents two tasks suitable for closer investigation: a Data Mining task, specifically Metamodel induction; and Agent Policy Optimization (APO), a newly defined Machine Learning task involving Automated Policy Generation within the previously discussed ED ABM. APO is ultimately selected for solving, using GP as a means to achieve policy evolution. In section 7.3, extensions to the ABM from Chapter 6 are discussed to make it amenable for APO. These extensions are mostly

changes in the agents which allow them to follow a policy which is programmatically determined at runtime. More details about APO are discussed in section 7.3, such as the policy goals of increasing patient flow and reducing infection spread. In section 7.4, a heuristic, denoted MASH, is introduced to help GP cope with some of the difficulties presented in 7.3, albeit the discussion is mostly task agnostic. Section 7.5 presents any remaining implementation details related to approaching APO with MASH-enhanced GP, including the specific fitness measure used. Section 7.6 discusses computational parallelism as a means to significantly speed up the learning process towards real time. This is followed in section 7.7 by the experiment, observations, and preliminary results of one such run of a Machine Learning task carried out within the ABM framework. Some conclusions are discussed in section 7.8. This author is the sole author of the following, yet to be published, work.

7.1 Application of ML to the ABM

For the reasons presented in section 2.5 Genetic Programming was considered and eventually chosen as the Machine Learning methodology for solving the APO task. According to [1], Genetic Programming (GP) is a Machine Learning methodology for the automatic induction of computer programs through an evolutionary process. GP is well established and includes successful applications in the areas of data mining [2], image classification [3], automatic circuit evolution [4], and online evolution of robot control [5], for example.

For the purposes of the following discussion, “model/abstraction” refers to a model/abstraction internal to the Artificial Intelligence (AI) or Machine Learning (ML)

``

system used. Thus the term "model/abstraction" is used to disambiguate such an internal model from the Agent Based Model (ABM) that has been presented throughout this thesis.

Often, the process for applying Artificial Intelligence (including Machine Learning), to a new problem domain involves the AI designer transforming the original problem through a model/abstraction into one that is amenable to solving by the AI designer's system. To contrast, GP solves the problem directly in the problem domain. Koza et al. [6] argue that many human-competitive AI successes demonstrate the intelligence and ingenuity of the programmer (or programming team) more than the machine, citing such examples as Deep Blue, IBM's world class chess playing system, as well as Chinook, the University of Alberta's "unbeatable" checkers program. The University of Alberta's human-competitive poker playing program, Polaris [7], could arguably be added to the list as well.

The healthcare system as a problem domain is an example of a system in which the underlying dynamics – being social in nature - are difficult to capture and highly nonlinear [8]. It is neither desirable or feasible for the AI practitioner to become an expert in each application domain they encounter. In light of this, GP emerges as a highly desirable candidate ML approach, because in the words of Koza et al. [6] GP has a high "Artificial to Intelligence ratio" – meaning that the value added by the GP method compared to the amount of intelligence supplied by the human (designer) applying GP is greater than the value added by other ML approaches.

Although a model, the ABM will be used in conjunction with GP, in terms of the Reinforcement Learning [9] approach referred to in section 7.2.2, the ABM is acting in

``

the role of the environment. This is clearly different from the internal model/abstractions constructed by human designers for the full problem in order to act as a sort of skeleton or template for the solution, and the ABM-ML hybrid system presented in this chapter uses no such internal model.

GP only requires that it can be provided with reasonable inputs, and that the quality of the output can be measured with respect to the problem definition. While there are other learning methodologies [10] available with this property, GP arrives at a program, or series of steps, that a person could interpret better than a set of numerical weights as in a Neural Network, for example. That sort of insight into how the Machine Learning methodology arrives at the solution could prove invaluable in the tasks considered in section 7.2; Automatic induction of a Metamodel [11] based on the ABM, and the APO task, or automatic induction (optimization) of agent policy within the ABM.

In the example of poker, one could consider whether it is of higher utility to have a solution consisting of hundreds of floating point numbers, meaningful only in the context of a program to interpret it, or a series of statements outlining a poker strategy, perhaps not as optimal as the first, but readable by humans.

While the following introduction to Genetic Programming is general and does not specifically relate to the ML task ultimately undertaken in this chapter, it provides a helpful context within which to understand the work. GP can be described as a stochastic beam search. In this case, the beam is the population of individual programs being considered. A summary of the basic GP algorithm is provided in Figure 7.1. A fitness evaluation executes the individual solution or GP expression tree within the problem

``

environment, and is used to identify the best performing individuals (individuals with the highest fitness). The specific meta-operators used for variation are described below.

```
Initialize current generation population of solutions with random solutions;
REPEAT:
    Evaluate fitness of each individual solution in the population;
    Construct next generation by reproducing highest fitness individuals;
    Apply variation meta-operators to individuals in next generation;
    IF termination criteria not reached
       current generation := next generation;
    ENDIF
UNTIL termination criteria are reached;
Top ranked individual by fitness is best performing solution found.
```

Figure 7.1. Basic Genetic Programming Algorithm

GP evolves computer programs in the form of functions. These are not pure functions in the Functional Programming (FP) [12] sense. A pure function only computes some value and returns the computed value without affecting machine state (such as the values stored in indexed memory). Side effects are used extensively, most notably for output and using indexed memory. For the following discussion, it is important to note that while mnemonics are used to identify functions such as “COPY”, this is for the convenience of the reader as the internal GP representation will use a single character symbol such as 'c' to represent “COPY”.

The evolved functions are organized into binary expression trees. They are binary because the operators used in the expression are limited to those with an arity of 2, that is they accept two arguments and return one value. All values are considered to be real (floating point). Operators operate on the return values from other operators and on values that come from primitive symbols belonging to what is called the terminal set. Such values are held temporarily in a lat-in-first-out stack data structure. A terminal

``

includes inputs, constants, and other no-argument (arity 0) functions. In a general terminal set independent of the choice of task, proposed inclusions are numerical constants of the integral values 0,1,..,9, fractional values 0.1, 0.2, ... , 0.9, and an arity-0 function to return a random number between 0 and 1. Others include a COPY() no argument function that returns the value at the top of the stack, or 0 if the stack is empty. The utility of COPY() only becomes apparent in relation to the NAND(X,Y) operator, discussed below. Transcendental constants such as Pi also will have apparent utility if a periodic function such as SIN(X,Y) is included. The application agnostic terminal set is summarized in Table 7.1.

Table 7.1 General (application agnostic) Terminal Set

Terminal(s)	Description
0,1,2,3,4,5,6,7,8,9	Numerical constants, integral values 0-9
a (RAND)	Returns a floating point random number between (including) 0 and 1.
b (COPY)	Returns the value at the top of the stack, thusly duplicating it
c (NEG1)	Returns -1.

Typically functions with arity > 0 are included in what is more commonly called the Function Set, but here it is referred to as the Operator Set (for clarity – to distinguish the Function Set from the set of functions being evolved to solve the task at hand). Basic arithmetic operators of addition, subtraction, multiplication, and protected division are also proposed for inclusion. Special care is taken to handle division by zero, hence "protected" division. In fact, special care must be taken to handle all possible values of arguments to an operator to avoid instability. Comparison operators of less than, greater than and equality are also desired. Since the work deals with real valued quantities,

``

'equality' shall mean 'within some predetermined tolerance'. If values greater than zero are treated as logically TRUE, addition and multiplication can approximate logical OR and AND respectively. NAND(X,Y) may also be included, since any boolean expression can be constructed out of NAND. COPY() is included to implement the logical NOT operation in conjunction with NAND, since NAND(X, X) = NOT(X). Take for example, a value Z at the top of the stack. The GP can represent NOT(Z) by first applying COPY() such that the top two stack elements hold the value of Z. Then applying NAND to the top two stack elements yields the result NOT(Z).

Another group of operators provides facilities for reading, writing to, and clearing indexed memory. A read operator returns the value of the memory at the index of the first argument and returns the second argument if that memory location was un-initialized (a sort of default). A write operator will write the value of one argument to a memory location of the second argument and returns the value that was written. Although this indexed memory mechanism is unsuitable for input as the inputs could conceivably be overwritten with the write operator, it is ideal for the output space. Of course, blocks of indexed memory do not have to correspond to I/O and can be used exclusively for storage. It can be noted that a GP with indexed memory (albeit of infinite size) and using an operator set including addition, multiplication, protected division, and subtraction, has been proven to be Turing Complete [13], that is, capable of computing any computable value. It is through the above-referenced operator and terminal sets that the GP has access to the program inputs and is how the program can perform output. Because side effects of functions are leveraged to such an extent, the operators LEFT(X,Y) and RIGHT(X,Y) are also proposed, in order to return X and Y respectively in order to allow the GP to

``

discard intermediate results that were used for a side effect operation such as writing to memory. LEFT and RIGHT might also be used as a cue to a human attempting to understand the evolved program, that the subtrees being discarded are executed earlier than and can be decoupled from the rest of the tree (as one goes up toward the root node). A summary of the proposed general (application agnostic) Terminal Set is found in Table 1, and a summary of the general Operator Set is found in Table 7.2.

Table 7.2. General (application agnostic) Operator Set

Symbol	Operator	Description
+	ADD(X,Y)	Adds the first argument to the second argument
-	SUBTRACT(X,Y)	Subtracts the second argument from the first argument
*	MULTIPLY(X,Y)	Multiplies the first argument by the second argument
/	DIVIDE(X,Y)	Divides the first argument by the second argument if the second argument is nonzero. If the second argument is zero, then return 1. [14]
<	LESSTHAN(X,Y)	Returns true if the first argument is less than the second argument
>	GREATERTHAN(X,Y)	Returns true if the first argument is greater than the second argument
=	EQUALITY(X,Y)	Returns true if the absolute value of the difference of the arguments is less than some arbitrary, predefined, threshold.
&	NAND(X,Y)	Considers arguments true if non-zero. Returns the first argument NAND the second argument.
%	MODULUS(X,Y)	X MOD Y, modular arithmetic operator.
‘	POW(X,Y)	Return X^Y
“	LOG(X,Y)	Return $\log_Y X$. Return 0 if any arguments are invalid.
$	WRITE(X,Y)	ABS then FLOOR is applied to the first argument, and the result is MOD-ed by the largest valid memory index to ensure that a valid memory location is accessed. Writes the second argument to the memory location indicated by the first argument. Return the second argument
@	READ(X,Y)	ABS then FLOOR is applied to the first argument, and the result is MOD-ed by the largest valid memory index to ensure that a valid memory location is accessed. Returns the contents of memory location indicated by the first argument. IF that memory is uninitialized return the second argument instead.
#	CLEAR(X,Y)	ABS then FLOOR is applied to the first argument, and the result is MOD-ed by the largest valid memory index to ensure that a valid memory location is accessed. IF the second argument is nonzero, set memory location indicated by the first argument to be uninitialized. Return the second argument.
(	LEFT(X,Y)	Return the first argument
)	RIGHT(X,Y)	Return the second argument

``

To implement these functions as binary expression trees using the discussed terminal and operator sets, two arrays can be used. One implements a stack that is used to hold temporary values of terminals and operator returns. The second array holds the GP evolved function or solution in postfix notation, that is, the operator follows the operands and thus there is no need for brackets in expressions. This second array is basically a character string that is read left to right; if a terminal is read then the value of the terminal is placed on the stack, increasing the stack size by 1. If an operator is read in then the top two values on the stack are taken to perform the operation, and the return value is placed back on the stack – a net stack size change of -1. An example of a GP solution string, its corresponding binary expression tree, and stack after each symbol is read is shown in Figure 7.2. This is a clever and well known representation [1], and as will be discussed in section 7.6 it is a convenient representation in conjunction with the MPI library for parallel computation.

``

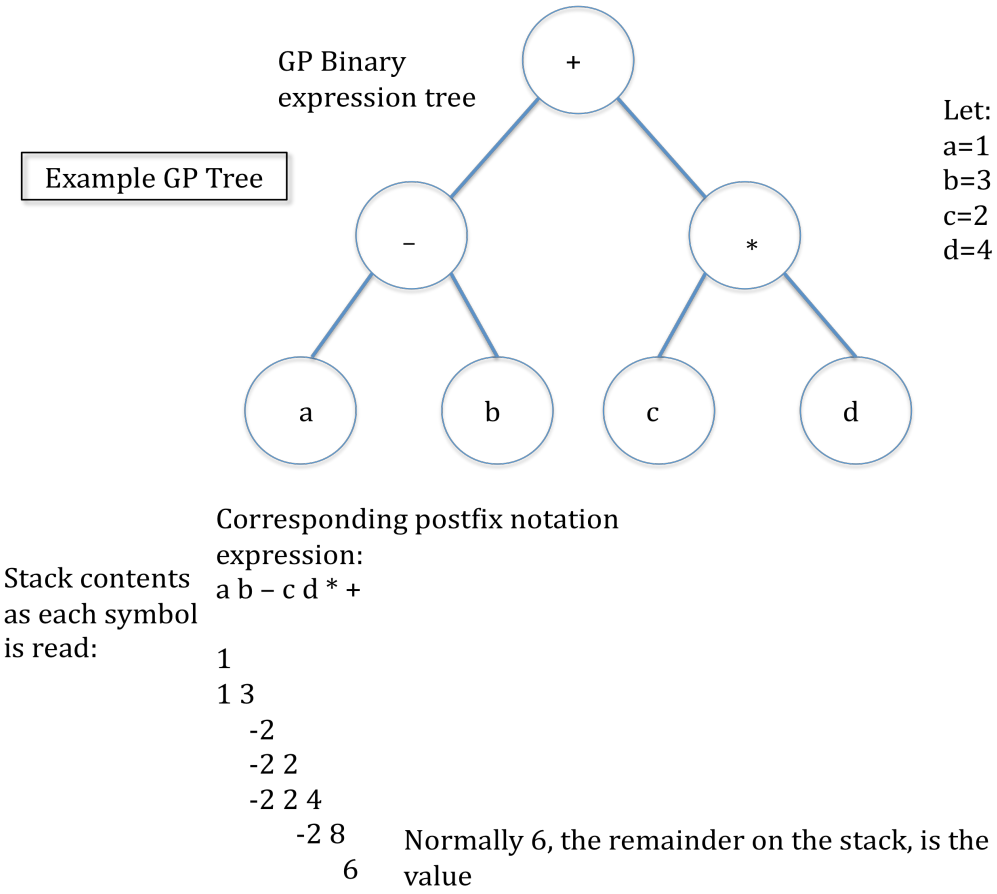

Figure 7.2. Binary Expression Tree, Corresponding Postfix Expression, Stack Activity

Not only is this representation very memory efficient, using one primitive character per symbol, but it is relatively easy to ensure that programs are syntactically valid. The conditions are: 1) programs begin with a terminal 2) the number of operands on the stack never drops below 1, so for all programs with length greater than 1, this means that the second symbol is also a terminal; and, 3) once all symbols are processed there is exactly one operand/value left on the stack. This final value is typically interpreted as the return value of the program, but as mentioned, the convention employed is to use indexed memory as output.

``

It is a conjecture of this work that restricting arity of functions to 0 and 2 is still representationally powerful enough to include any interesting and useful functions that may be required. In case a function of arity 1 becomes absolutely necessary, a dummy argument can be used. By making this restriction, the complexity of the GP interpreter and internal representation can be reduced greatly.

There are five meta-operators that operate on GP solutions (p_i) that are used to transform the population of one generation to the population of the next generation. As referenced previously, the selection operator which chooses p_i for the remaining four meta-operators, usually based on the performance of p_i with respect to the fitness function. The p_i selected to be in the next generation are replicated by the replication meta-operator. Typically, replication is also accompanied by crossover and mutation. Crossover is a biologically inspired operator that mimics genetic exchange in sexual recombination. Selection is used to find a mate in the population for the replicated p_i , then subtrees are exchanged between these two p_i . An illustration of crossover is in Figure 7.3; note that the subtrees chosen for crossover do not have to be from structurally similar parts of the expression tree. Mutation is an operation that makes some random changes or perturbations to the replicated p_i . A point mutation is depicted in Figure 7.3, but one should note that there are other types of mutation. Deletion is implied for any p_i not selected for replication into the next generation. As mentioned, restricting operators to an arity of 2 makes it much easier to ensure that the outcome of crossover and mutation is a syntactically valid expression tree. The author of [14] in general only uses crossover as a means of perturbation, while [1] lists a myriad of different mutation operations of which only point mutation has been discussed. The combination of arity

``

restrictions and the array encoded tree representation make implementing all the GP variation meta-operators in [1] feasible even by a single programmer. The variation meta-operators implemented in the GP system discussed here are: collapse subtree mutation, permutation, point mutation, expansion mutation, hoist and gene duplication. Additional details can be found in table 9.2 of [1]. As is suggested in [14], when an individual is selected for replication into the next generation, it undergoes the crossover operation with another selected individual 90% of the time. In the remaining 10% of the time, one of the implemented mutation operations is chosen with equal probability.

It is suggested in [1] that during a GP run individuals become inordinately long, filling up the majority of their length with non-functional symbols called introns. This phenomenon, called bloat, emerges because useful parts of the solution are protected from destructive crossover and mutation events because a random crossover or mutation point in an individual tree riddled with introns will likely hit an intron. Because introns are by definition non-functional, such a mutation or crossover event cannot negatively impact the fitness of the individual. One way that introns can manifest is a lengthy expression that is multiplied by 0 which is then added to a functional subtree. The result is entirely dependent on the functional subtree. Therefore, this work claims a per-symbol mutation or crossover rate may be beneficial. Longer individuals will experience more mutation and crossover events in general than more concise individuals, removing the emergent selection pressure towards longer individual solutions. To counter the tendency of certain GP variation meta-operators, such as subtree crossover, to increase the length (number of symbols) in individuals, a meta-operator called “trim” is introduced. Trim is applied after the variation meta-operators previously mentioned increase the length of a

``

subtree to greater than 80% of the maximum allowable subtree length. Trim repeatedly chooses one of the variation meta-operators that tend to reduce subtree length (eg. subtree collapse, hoist) until the length is less than 80% of the maximum allowable.

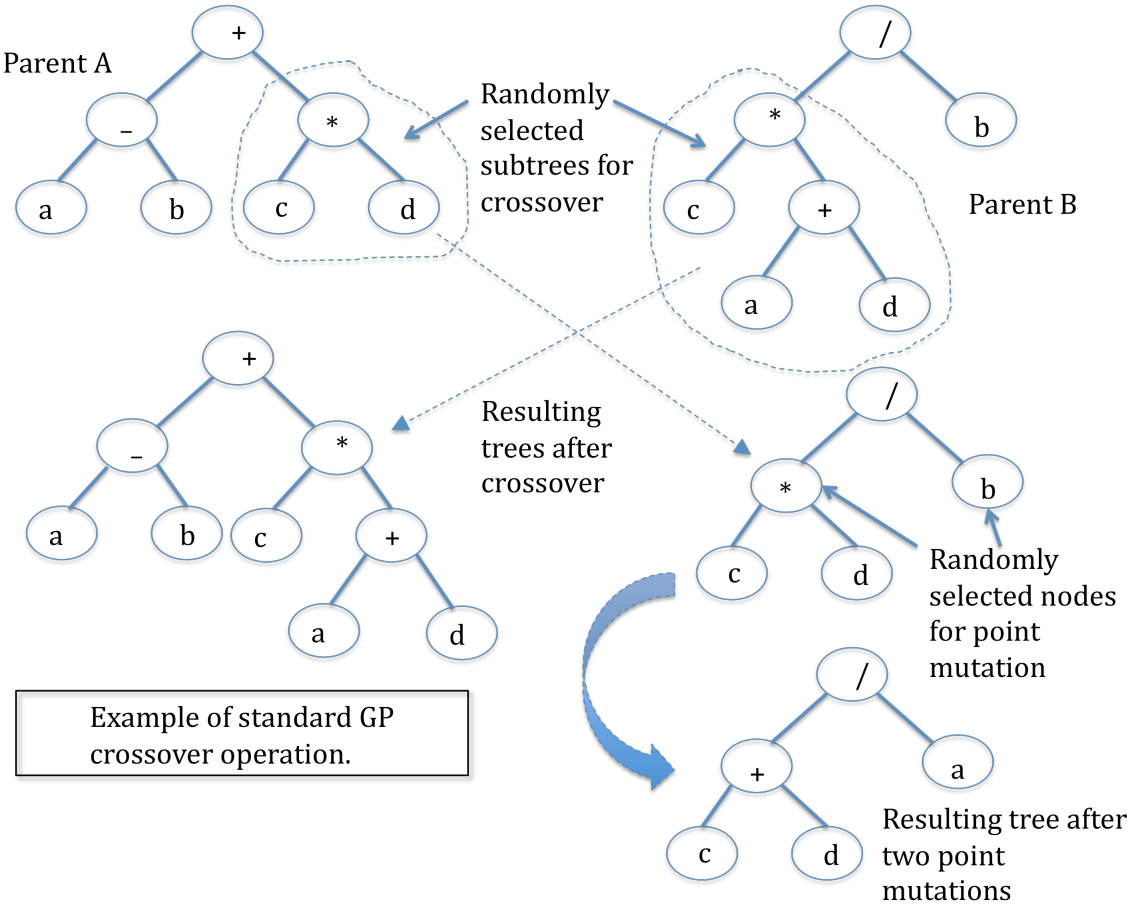

Figure 7.3. Example of Crossover and Mutation on Arbitrary Binary Expression Trees

Further reading on the topic of Genetic Programming, can be found in [1].

7.2 Machine Learning Tasks

This section presents two potentially useful applications of Machine Learning in conjunction with the Emergency Department ABM in the greater context of a decision

``

support tool for healthcare practitioners. The first application, data mining, has several specific ways of supporting the ABM. The second application defines the basics of the Agent Policy Optimization (APO) Machine Learning task with the goals of increasing patient flow, and reducing spread of the Influenza Like Illness causing virus presented in Chapter 6. Ultimately, the latter application was chosen, as it represents a unique opportunity for novel work, and it is also the more interesting of the two applications.

7.2.1 Data Mining Applications

Data Mining can be performed in support of the ABM either operating on the input to the ABM, the output of the ABM, or both. Chapter 6 presented an analysis and decision process in which much of the data analysis was performed by hand – representative of expert opinion and best effort policy making.. In the context of a decision support tool, it is desirable to automate some if not all of this analysis. If presented with raw data and the desire to incorporate this data as an input to the ABM, it may be valuable to perform a symbolic regression [15], to better understand the input data. Similarly, in the course of analyzing ABM output, a symbolic regression may be performed on the output.

A more difficult task would be to automatically generate a metamodel (an abstracted model of the original model) from the ABM. The following discussion concentrates on metamodel induction as the data mining goal, but the same general approach applies to the aforementioned symbolic regression. The authors of [16], although they used a Neural Network to form a metamodel of their ABM, found that once trained, the metamodel was useful for providing interpolated results between parameter spaces that were sampled using the ABM. The metamodel was much faster at providing results for

``

these interpolated cases. The advantage of using GP would be that the result of learning would be a symbolic description of the original ABM, whereas a Neural Network results in a potentially complex table of weights. Should data from the data collection framework posited in Chapter 3 and Appendices A and B become available, the aforementioned ML system could be trained and validated using real data to predict conditions in the Emergency Department (ED). Such a successful predictive data mining application would provide immediate ability for the data collected to be usefully mined. Accuracy aside, any such forecast would have to be available before any analysis where a human analyst was involved. Arguably this is a simulated predictive data mining task, here the 'data' is generated by the ABM simulator, the research however positions the ML within the context of real data being available from the ED in real time.

Specific examples of features of the ABM and underlying system that can be modeled/mined by the GP include:

- Forecast throughput or conversely wait times for service at an ED as a function of patient arrival rates, staff levels, and current number of patients of each class already in the ED system.
- In the infection spread extension, forecast infection rates or "potential" as a function of patient arrival rates, staff levels, initial infection rates, immunity rates, and current number of (infected) patients of each class already in the ED.

Figure 7.4 illustrates how the ABM provides feedback to the GP-based ML system for the Metamodel Induction task. To accomplish this, the GP begins by generating thousands or perhaps millions of candidate solutions in the form of computer programs that predict the desired ED feature (e.g. forecast of waiting time for an arriving patient)

``

for a given ED state. Let P be the current set of n solutions, also referred to as the population. The form of these solutions is discussed below. The ABM is brought to some state S_t for which the set V_t of k state variables v_{t0}… v_{tk} is recorded. V_t describes the state of the simulated ED at time t. Examples of state variables include: patient arrival rates, staff levels, and current number of patients of each class already in the ED system for wait time forecasting, as well as initial infection rates and immunity rates for infection spread forecasting.

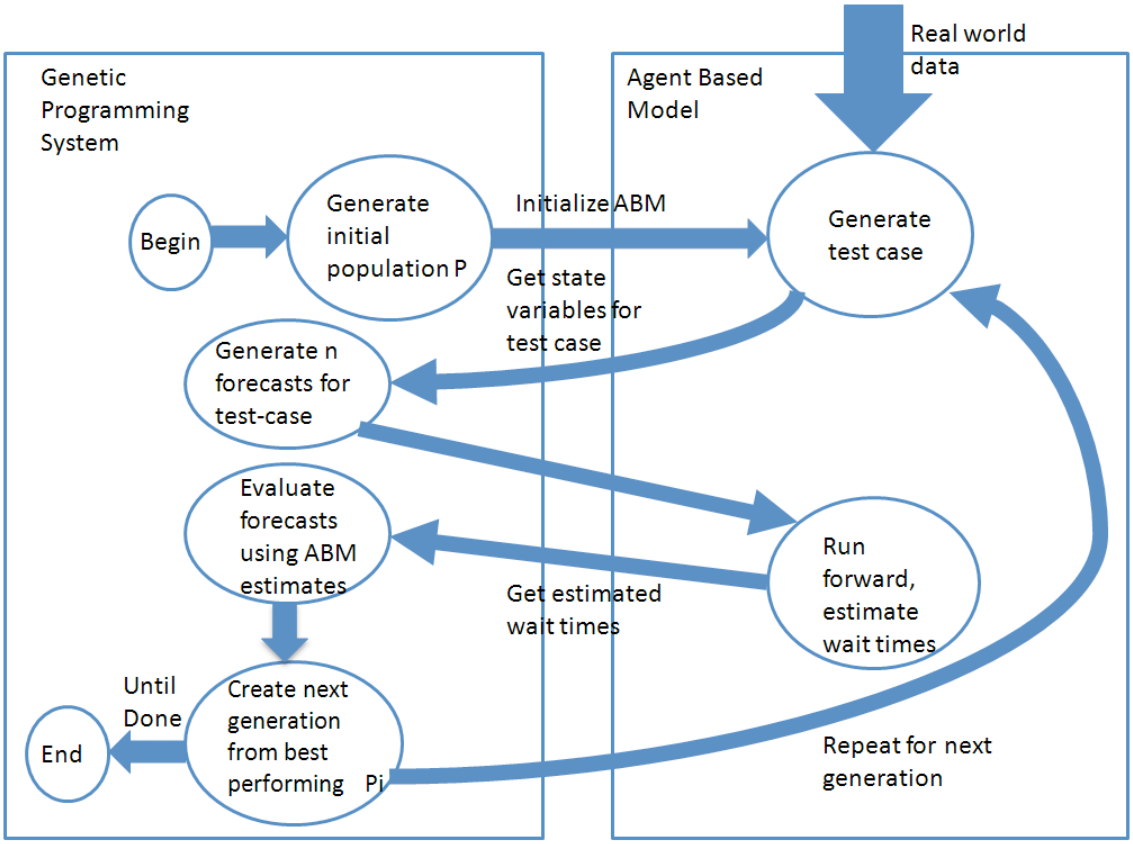

Figure 7.4. Hybrid Genetic Programming, Agent Based Modeling System Interaction for Metamodel Induction Task

``

The GP-system has an interpreter that generates a set of outputs, $O_i = p_i (V_t)$ where $p_i \in P$, for each program, p_i, in the population, P, by executing the program with V_t as input. The outputs, O_i represent a forecast for time $t + \Delta$ for a predetermined feature of the ABM. For example, forcasted features may include the expected wait time for each patient class, or the propensity for infection spread given some ED state. The remainder of section 7.2.1 discusses forecasting patient wait times for each patient priority class. It should be noted that for this type of task, the ABM simulator will have to be modified to expose each of V_t to the GP-system at the appropriate time. Each O_i has m elements, in this case, one forecast for each patient class of interest. The ABM is then run forward, perhaps many times with different random number seeds to get an average, to observe wait times experienced by the patients in the ABM. The solutions in P can then be ordered based on how well their forecasts match those generated by the ABM (Mean Squared Error is one possibility). This evaluation is known as a fitness function. The best solutions in P according to this criterion are probabilistically selected to form the basis of the population of the next generation. This process is then repeated for the next-generation population until some criteria is reached. According to [1], overfitting (where the ML system fails to generalize from the training data to real-world data it has not encountered) can be mitigated (or at least detected) with the following technique. Available data is partitioned into three sets: the training set, test set, and validation set. The GP system is trained on the training set, until some measure of performance (fitness) is reached. One measure for finding the most generalized solution out of the population would be to evaluate every individual in the population using the instances in the test set, which the population has never been exposed to. The best performing individual on the

``

test set can be argued to be the individual that generalizes best. The accuracy of this "best generalizer" individual is then estimated using the third, validation data set. Therefore, a run can perhaps be terminated when the performance of the population continues to improve on the training set, but levels off or drops on the test and validation sets – in which case overfitting is presumed to be taking place.

The search space is explored because new solutions generated from the best performers of the previous generation will always have some variation introduced by special meta-operators discussed in more detail in a later section. The quality of solutions with respect to the fitness function will tend to improve over the course of generations because the best performers tend to be selected or replication into the next generation. For a pseudocode description of this iterated learning process, see Figure 7.5. Clearly, it would be desirable to train and validate the wait-time forecasting system on real data; the ABM here is only acting as a data generator until such data is available. Such a data driven system would be a classical example of using supervised learning to perform predictive data mining [15].

``

```
Begin
Generate original population P
Repeat
  Run ABM up to time t
  Record state of ABM at time t, V_t
  For each p_i in P
     For each patient class
        Generate O_i from p_i, V_t
     End For
  End For
  For each patient class c in set of patient classes
     Repeat using different random seeds
        ABM generates patient arrival of class c
        ABM simulates how long generated patient waits
     Until satisfied with sample size
  End For
  Rank each p_i based on how close O_i is to ABM output
  Construct population N from numerous top-ranking p_i
  P:=N
Until termination criteria are reached
Highest ranked p_i is best predictor.
End
```

Figure 7.5. Pseudocode for Forecasting Task

It should be noted that training data can come from sources other than the ABM: the interface to the ABM is simply switched for an interface to a file or database. This is made possible by the object oriented design of the system, and highlights the flexibility of the approach. As seen in from Figure 7.6, there is little difference whether the data comes from the ABM, or some other source; only the labels on the right hand side of the diagram have changed signifying that although the source of the data has changed, the program flow is virtually identical. The setup described in Figure 7.6 would be sufficient for symbolic regression without significant modification.

``

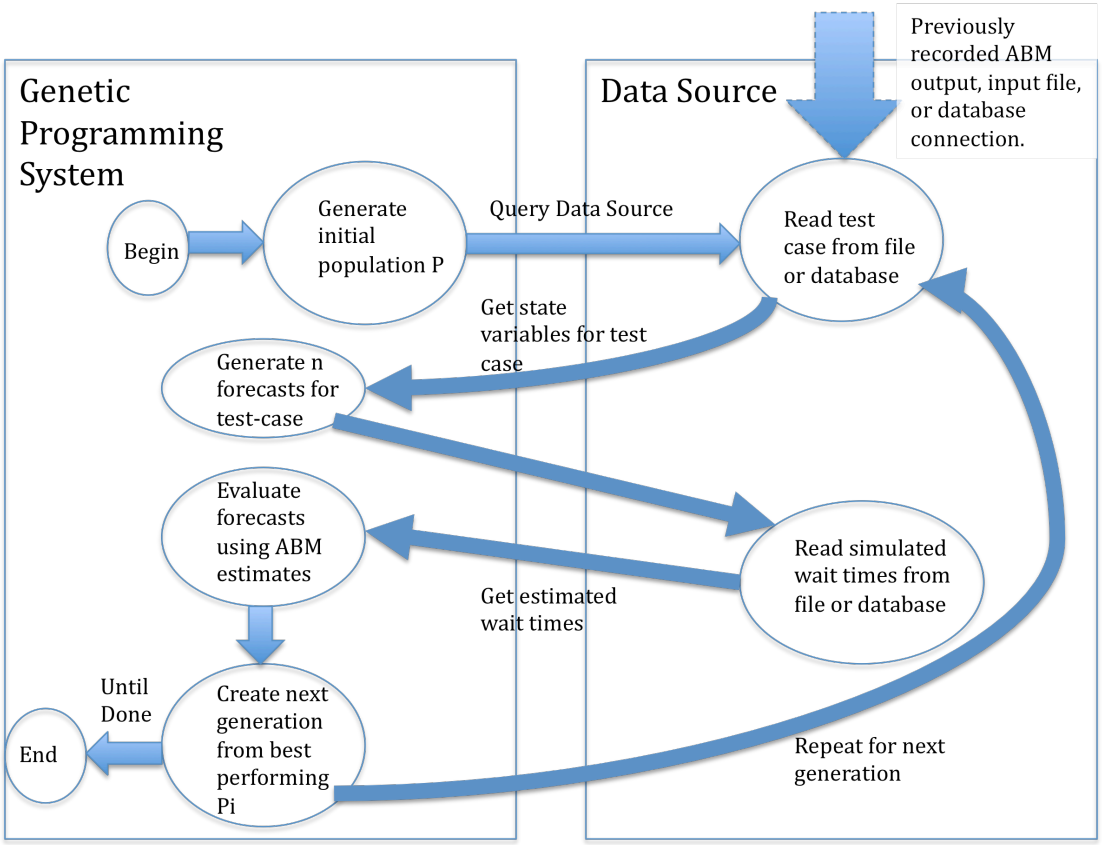

Figure 7.6. A more traditional mode of data mining.

7.2.2 Agent Policy Optimization

Discovering or generating optimal or near-optimal agent policies is the second application of ML within an ABM based decision support tool for healthcare considered in this thesis. The automated evolution of agent policy in the context of Agent Policy Optimization is a more ambitious goal than the data mining related Metamodel induction task outlined in 7.2.1, and it is the application chosen to implement and focus on for the remainder of this chapter. APO offers greater opportunity for novel results compared to data mining, as a Metamodeling task was already explored in relation to a more granular DES based ED model in [16]. For now, a brief introduction to the Agent Policy Optimization task is provided.

``

Various policies could be evolved within the ABM, however, in order to have the greatest impact on the operation of the modeled ED, the evolved policy should have the goal of mediating interaction between service provider (doctor, nurse) agents and service consumer (patient) agents. This interaction is portrayed in Figure 7.7. Since this application will be based on the ILI causing virus spread simulator discussed in Chapter 6, the specific goal of the APO task will be to concurrently maximize patient flow and minimize the number of patients acquiring the ILI causing virus during their time at the simulated ED. As would be the objective of policies for an ED during an ILI pandemic or epidemic outbreak.

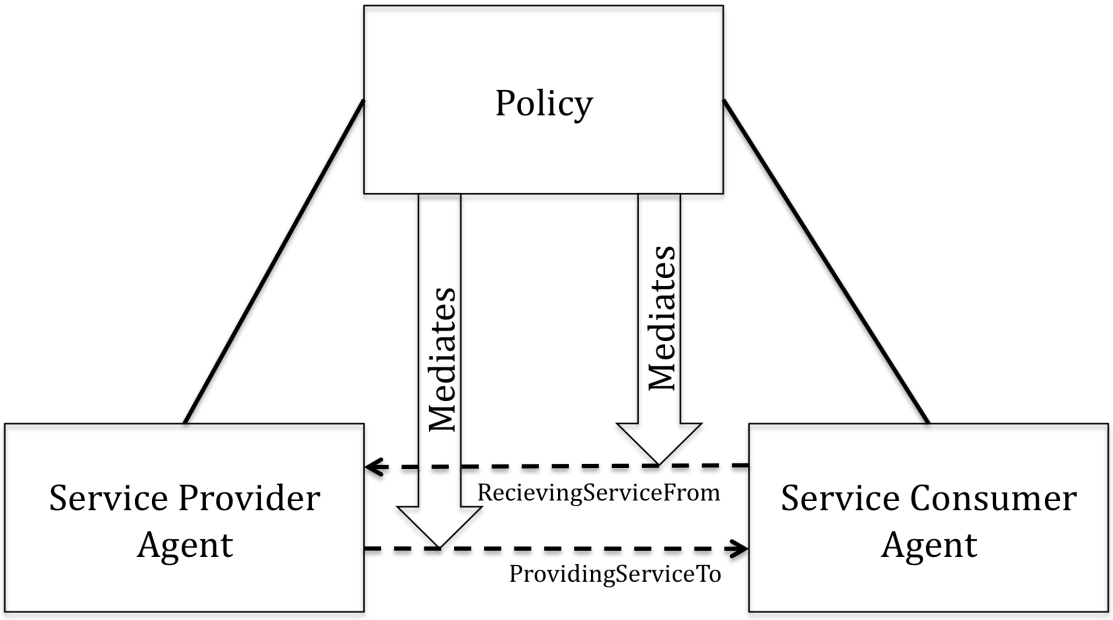

Figure 7.7. A policy to mediate interaction between service provider and service consumer agents.

In contrast with the Metamodel induction task which used a supervised learning approach, this work most closely follows (on-line) Reinforcement Learning (RL) as the Machine Learning approach for the Agent Policy Optimization task [9]. The prototypical

``

application for RL systems is agent policy determination in a Markov Decision Process (MDP), a class of problems in which the state s of the environment is known at time t, and the agent takes some action a, and receives a reward r(s,a,t) corresponding to the desirability of taking action a in state s at time t. In a Partially Observable MDP (POMDP), the state of the environment can only be partially determined; that is, with some uncertainty. As will be discussed, the Agent Policy Optimization task has some inherent component of a POMDP. Similarly, the correct output sought by the ML system may not be known, however, some feedback signal concerning the quality of the solution being evaluated is available.

Related work within the field of healthcare informatics includes an example of evolving staffing schedules using a form of Evolutionary Algorithm [17]. There are numerous examples of co-evolving [18][19], or distributed learning [20], of strategies in a multi-agent domain, albeit not healthcare specific. Unlike this latter example, the evolved policy is housed in one agent, the ErController which in a real-world "incarnation" might be manifest as a data terminal accessible information system, deployed in the ED. Such a deployment with decision support capabilities might be able to provide direction to ED staff based on the real-time state of the ED, thereby effecting its policy of coordinating the ED.

As discussed previously, GP can improve upon human generated results, sometimes in unexpected ways [6][14]. Therefore, such an automated policy generation system would have a role in part of a greater policy decision process.

The discussion in previous chapters suggested a means for using the ABM of the ED for testing ED policy before implementation in the actual hospital ED. However, this

``

requires a human to generate a policy for the ABM to test. Examples of such policies are "staff with four doctors instead of three", "begin to divert patients once the number of waiting patients exceeds a particular threshold", or "place ILI symptomatic patients in an alternate treatment stream". The work now focuses on a system in which the GP automatically generates policies, uses the ABM to evaluate these, and uses the best individual policies as a basis for the next generation of policies. Specifically, these policies are in the context of the APO task as defined in this chapter.

```
Begin
Generate original population P
Repeat
   r:= generate random number
   for each Pi in P
      initialize ABM replication using r
      ABM executes Pi
      Record overall wait times
   End for
   Rank each Pi based on fitness
   Construct population N from group of best Pi
   P:=N
Until termination criteria reached
Top ranked Pi is best performing policy.
End
```

Figure 7.8. Pseudocode for APO Task

This process is then repeated many times until pre-defined criteria are met. An example of such terminating criteria is several generations passing without any performance improvement in the population according to the fitness function. Another example is terminating the run when the average length of individuals (symbol count) increases significantly without a corresponding improvement in average fitness [1].

For reasons that will soon be apparent, differences in the APO task make it much more computationally intensive. For the forecasting task, the state of the ABM was unaffected by O_i , the output of the GP interpreter. The ABM independently generates

``

state variables V_t , and corresponding wait times. Therefore, one run of the ABM is sufficient to test all p_i. In contrast, for the Agent Policy Optimization task, each individual policy p_i must be evaluated from within the ABM. O_i is meaningful only in the context of the ABM and is immediately consumed by the ABM agent as it executes the policy. Furthermore, the state of the ABM at time $t > 0$, S_{at}, resulting from executing policy p_a can be expected to be different from the state S_{bt} , had policy p_b been executed instead. Each p_i should be evaluated with respect to a set of initial conditions or patient flows in order to capture the average behaviour of the p_i on a variety of patient flows. The aggregated performance of p_i over the set of patient flows represents the fitness of individual p_i. Let k represent the number of random seeds, r_j,where j={0..k} (each used to generate a unique patient flow) that is reasonable given the population size, n, and the available compute resources. A separate ABM replication needs to be run for each combination of p_i and r_j to be evaluated. Therefore, n*k ABM replications need to be run every generation. That is, each p_i needs to be evaluated by an independent ABM replication once for each random seed. The same set of k random seeds should be used to evaluate each p_i to reduce the effect of chance.. A different set of random seeds should be used each generation in order to maximize the number of training samples (patient flows) observed by the learning system. It is likely that this fitness evaluation will have to be done in parallel on a compute grid as a result of the computational intensiveness of fitness evaluations for the APO task.

One reason for using the Reinforcement Learning style of ML is the number of infections or patient wait times for the optimal policy given a particular patient flow is unknown. In fact, the form of that policy is unknown. The only feedback GP gets is the

``

preferential order of policies with respect to one another as defined by a fitness function. Depending on the application, the fitness function may assign a higher utility to policies that result in shorter waiting time. Alternately, the fitness function may assign a higher utility to policies that result in fewer infected patients. Multiobjective fitness functions are made possible by combining two or more features, for example summing the respective fitness scores for each feature. Figure 7.8 shows pseudocode for this process, and a diagram of the process is in Figure 7.9. Here, using an ABM to evolve patient diversion or triage policy is desirable, because doing so in an actual ED would require considerable time to evaluate, and also more importantly has ethical implications as it risks actual patient care. It is interesting to note the parallel between the evolutionary approach of the GP, and the evolution of policy in a real ED (successful polices tend to persist in order to be incrementally improved upon). In other cases, a significant improvement is seen within an ED when a major change is undertaken. This is resonant of evolutionary processes where gradual anagenetic improvements are punctuated by major events.

As with the Metamodel induction task, the ABM simulator will have to be modified so that the GP has access to V_t. Once again, this includes things such as patient arrival rates, staff levels (number of doctors), and current number of patients of each class already in the ED system for wait time forecasting, as well as initial infection rates and immunity rates for infection spread. In addition to these, an ErController targeted for Agent Policy Optimization needs to be provided with the type (doctor, patient, nurse) of the current agent for which it has to make a policy decision. In the infection spread

``

ABM, a partially observable indicator of whether the arriving patient is infected should also be provided as input, that is, there is a probability of misdiagnosis.

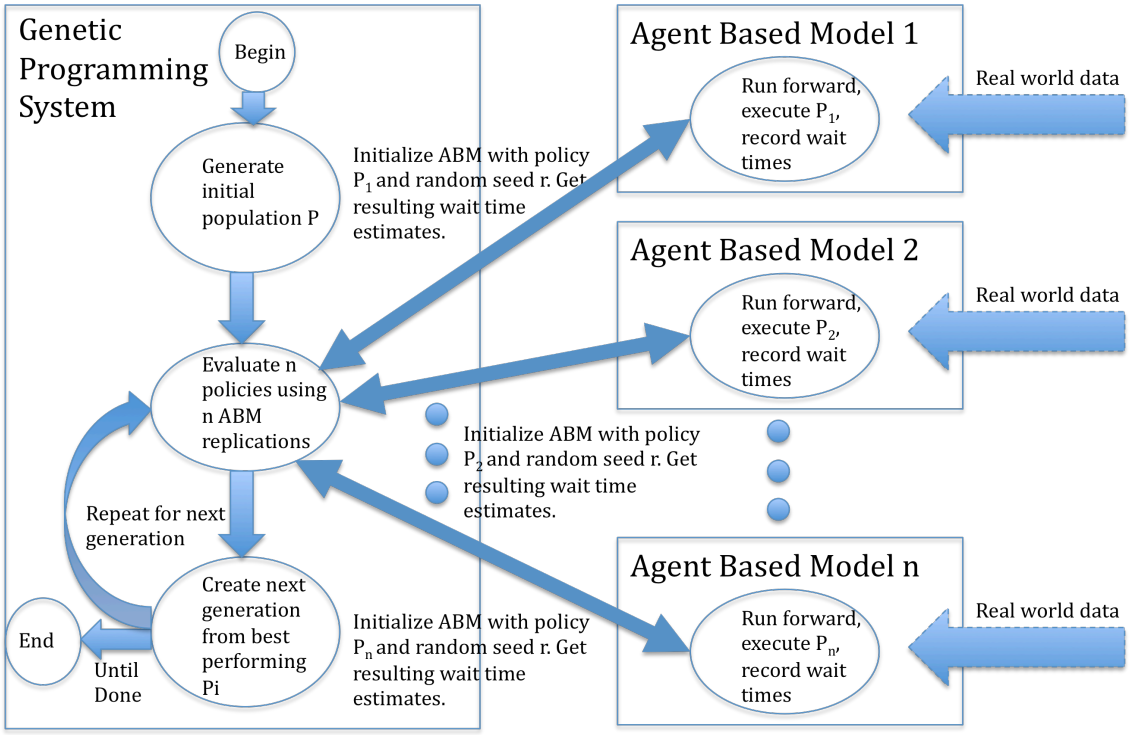

Figure 7.9. Hybrid Genetic Programming, Agent Based Modeling System for APO Task

Such an investigation would also require extending the ABM's ErController agent to include an interpreter capable of executing policies generated by the GP-based system. As discussed earlier, O_i is generated within the ABM agent executing the GP-generated policy, and O_i is consumed immediately within the context of the ABM. For Agent Policy Optimization for the triage agent, the output may indicate to which treatment queue to assign an arriving patient, or in the case of the infection spread ABM, which interventions to implement for this patient, if any.

Another possible related application for which policy could be automatically generated is a patient diversion policy among numerous EDs in a health region. A similar

``

patient diversion scenario is explored in Chapter 3, but appears to be of lesser interest to the Healthcare Informatics and policy communities, however. Such a policy could be said to be architecture-agnostic because only the policy itself is evolved; the means of implementing the policy is assumed to be in place and is treated as abstract. It would be beneficial to optimize patient diversion across all EDs in a region, rather than promote competition, all EDs will share the same policy. Such a policy will be arguably more generally applicable than a set of unique policies evolved for specific modeled EDs. An agent class responsible for executing patient diversion policies generated by the GP-system would have to be added to the ABM.

The preceding discussion on the implementation of Agent Policy Optimization (policy evolution via GP) has remained intentionally abstract. Concrete details are further developed in sections 7.3 which outlines the extensions made to the ABM since Chapter 6, necessary to perform the APO task; section 7.4 discusses extensions to the basic GP version presented in section 7.1, in order to strengthen the representational characteristics of GP for the APO task; and section 7.5 explains the combined, final, ABM with GP for APO within a decision support context.

7.3 ABM extensions for Agent Policy Optimization

Several improvements to the ABM as presented in Chapter 6 need to be made so it is capable of executing a policy determined at runtime, which is a requirement for the APO task introduced in section 7.2.2. The functionality of the ED before triage is complete

``

and is virtually identical to the ABM in Chapter 6, with the exception noted in 7.3.2. The ErController agent has had its functionality expanded so that it is capable of executing the GP generated policy at run-time. The ErController typically interacts with the agents by assigning them a room and a priority. The doctor agents and patient agents have had their state transition rules modified in order to work with the new ErController. A transfer nurse agent was added, which extends the ABM to place a natural limitation on patient transfers. If the means of transferring patients between the waiting and treatment rooms was left abstract, the ErController would likely be able to order more patient transfers between rooms than would be possible in a real ED. Finally, various software framework hooks were inserted into the ABM in order to allow the GP module to determine the current internal state of the ABM. The specific ABM features which are ultimately queried depend on the terminals and operators specified in section 7.5.

7.3.1 Simulated Agents Involved in APO

Until triage is complete, the patient functions identically to patients in previous iterations of the model. In the current version, after triage is complete, the ErController assigns the patient a waiting room as well as a treatment queue priority. The patient enters the "Wait for Treatment" state, in which it will wait in a waiting room, until the patient is transferred to a treatment room. If the ErController assigns the patient a treatment room immediately, the request to move the patient is added to the transfer queue, but until transferred the patient will wait in a waiting room. At any time, the ErController or a doctor may order a patient to be transferred to another treatment or waiting room, pending the availability of the transfer nurse. When the transfer nurse arrives to transfer

``

the patient, it puts the patient into the “Transfer” state in which the patient will follow the transfer nurse to the destination room which could be a treatment or waiting room. Once the transfer is complete the patient returns to the “Wait for Treatment” state. If the patient is in a treatment room, based on ErController policy, the doctor may eventually arrive to treat the patient. When treatment begins, the patient is removed from the treatment queue, and the patient enters the “Receive Treatment” state. Treatment continues until either the patient’s treatment is completed and the patient leaves the ED, or the ErController interrupts the treatment. If the treatment is interrupted, the patient is reassigned to a treatment queue. Figure 7.10 is a state diagram summarizing this behaviour, with Patient-ErController interaction highlighted in red.

``

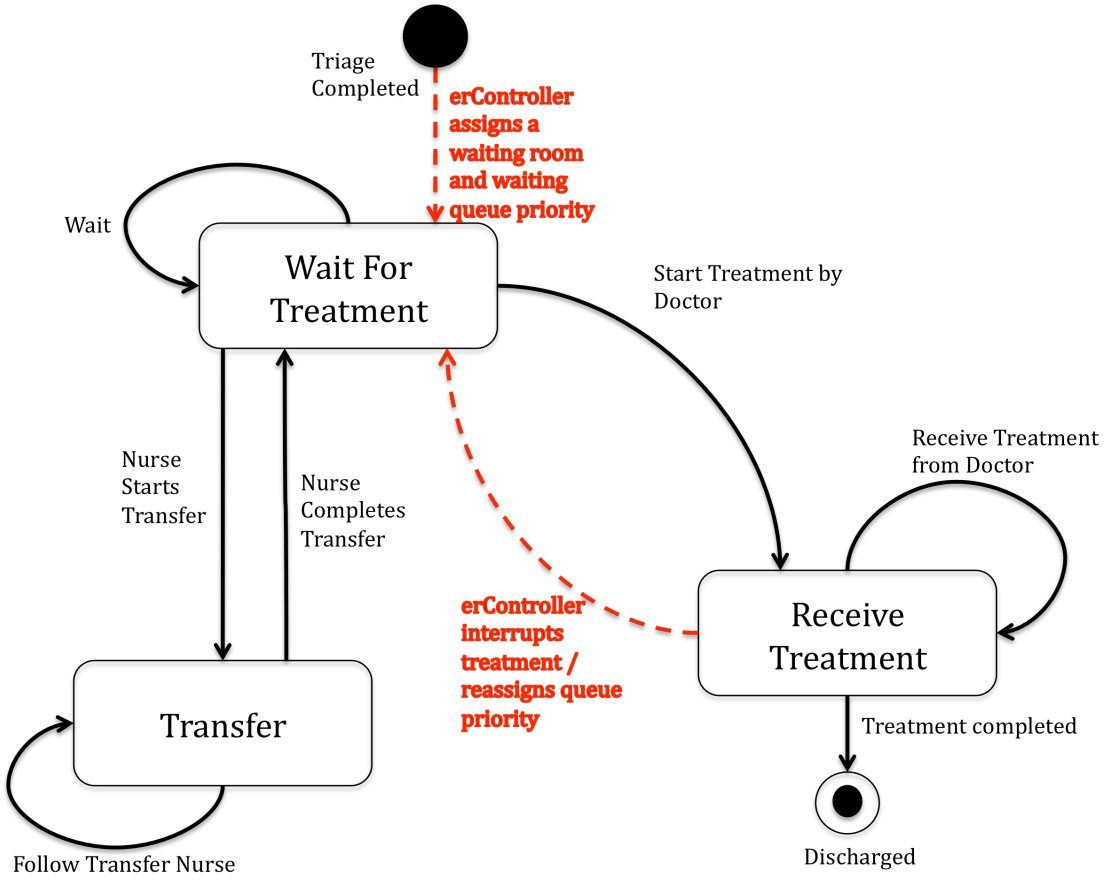

Figure 7.10. Partial State Diagram of Patient after Triage, interactions with ErController are in red.

The doctor begins in the "Idle" state, and from this state, each time step requests a room and treatment queue priority from the ErController. When the ErController assigns the doctor the room and treatment queue priority, the doctor transitions into the "Assigned" state. In this state the doctor proceeds to the assigned room. Upon arriving at the assigned room, if there are no patients in the room, or there is already a doctor treating a patient in that room, the doctor will transition back to the "Idle" state in order to get reassigned to another room. Otherwise, the doctor chooses a patient inside the assigned room closest to the assigned treatment priority, transitioning that patient to the "Receive Treatment" state. If there are no other patients in the room, the doctor

``

transitions to the “Treating” state and asks the ErController to have all patients besides the chosen patient transferred to other rooms. The doctor then waits until the room is cleared of all patients other than the chosen patient before transitioning to the “Treating” state. At any time the ErController may notify the doctor to interrupt the treatment of the current patient, which will cause the doctor to transition back to the “Idle” state for reassignment, and the patient transitions back to the “Wait for Treatment” state described in Figure 7.10. A summary of the doctor agent’s states is presented in Figure 7.11.

The transfer nurse agent begins in the “Idle” state, requesting, and waiting for an assignment from the ErController. From here, the transfer nurse will behave differently depending on whether the transfer queue is empty or not.

In the first case, when the transfer queue is empty, the ErController will assign the transfer nurse a *room* and a treatment queue priority and the transfer nurse transitions to the “Assigned to Room” state. In this state, the transfer nurse proceeds to the assigned room. Upon arriving, if the transfer queue is no longer empty, or there are no patients in the assigned room, the transfer nurse transitions back to the “Idle” state. Otherwise, the transfer nurse signals for the ErController to order a transfer for the patient in the room whose treatment queue prioirty most closely matches the assigned treatment queue priority. The transfer nurse will then transition back to the “Idle” state.

In the second case, when the transfer queue is *not empty*, i.e. the ErController has ordered a transfer, the ErController will assign the transfer nurse a patient and transfer destination based on the head of the transfer queue. The head of the transfer queue is removed from the queue, and the transfer nurse switches to the “Assigned a Patient” state. In this state, the transfer nurse proceeds to the room occupied by the assigned

``

patient. Once the transfer nurse arrives, if the assigned patient is being treated by a doctor, the transfer nurse will cancel the transfer and return to the “Idle” state. Otherwise the transfer nurse enters the “Transferring” state, and the assigned patient enters the “Transfer” state. In this state, the patient will follow the transfer nurse to the transfer destination room which may be a treatment or waiting room. Once both agents have arrived at the transfer destination, if the destination room is full or there is a doctor treating a patient in the destination room, the transfer nurse chooses another transfer destination that is not full and does not contain a doctor treating a patient; the transfer nurse remains in the “Transferring” state and proceeds to the new room. Otherwise, if the room is not full and there is no doctor treating a patient, the transfer is considered successful. The transfer nurse returns to the “Idle” state and the patient returns to the “Wait for Treatment” state. Again, patient state transitions can be found in Figure 7.10, and the transfer nurse agent’s state transitions can be found in Figure 7.12, with ErController interactions highlighted in red.

``

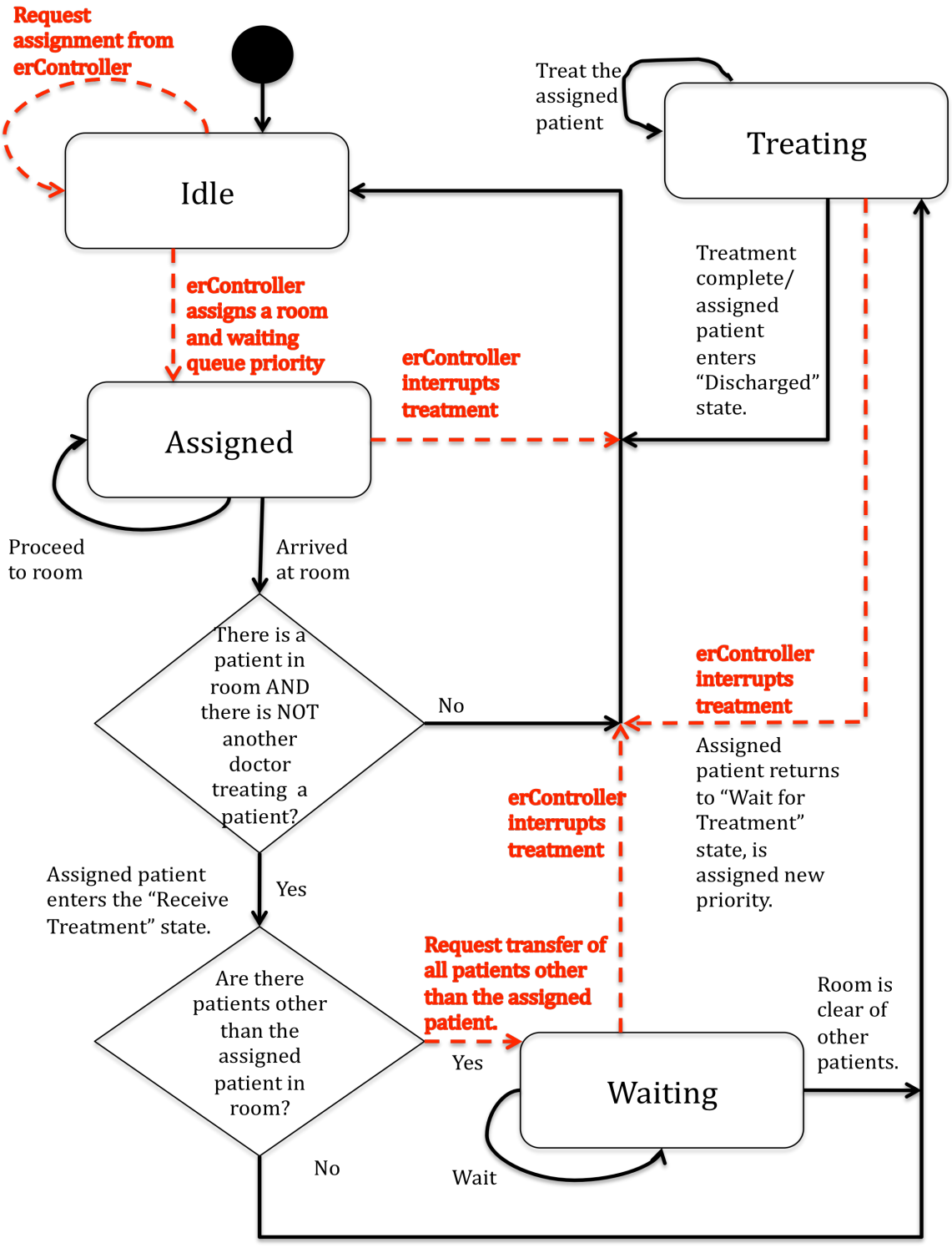

Figure 7.11. State Diagram for Doctor Agent, interactions with ErController are in red.

``

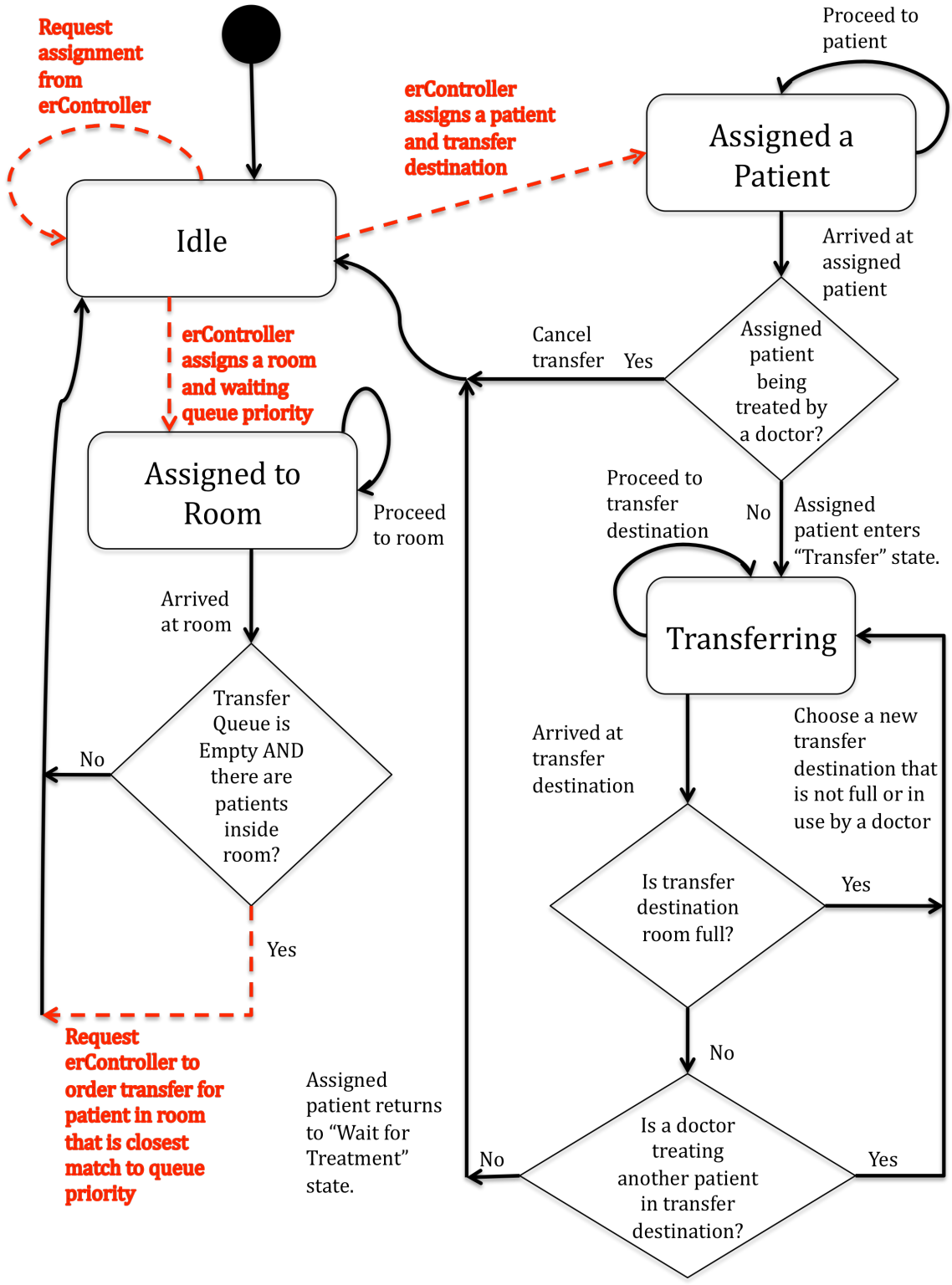

Figure 7.12. State Transition Diagram for Transfer Nurse Agent, interactions with ErController are in red.

``

The role of the ErController is a support system that directs ED processes. Therefore, for a given state of the ED, and for a given agent, the ErController directs that agent to take some action. The ErController must administer queues, and manage room assignments to minimize ILI causing infection spread, and to maximize patient flow - this is essentially the APO task. These problems have been shown to be related in Chapter 6. The ErController may use treatment rooms as temporary waiting rooms, but waiting rooms cannot be used as treatment rooms because they are not properly equipped. A summary of inter-agent interactions as well as collaboration between agents and other simulated entities that are mediated directly or indirectly are shown in Figure 7.13.

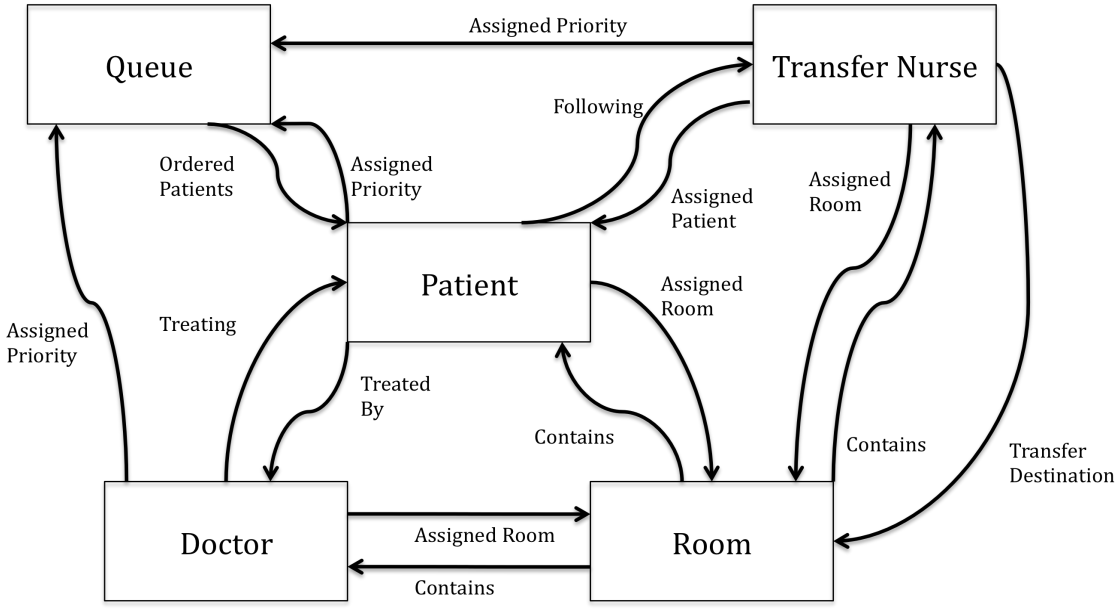

Figure 7.13. ABM relationships mediated by ErController's evolved policy.

7.3.2 ABM Assumptions and Constraints for Agent Policy Optimization Task

As mentioned in the previous section, the ErController agent behaves as a support agent for coordination and control of ED processes. In this implementation, the ErController queries the GP evolved policy whenever there is a decision to make, therefore, the evolved policy is a centralized one. However, as mentioned earlier, it is possible to co-evolve strategies between cooperating agents. Although the examples cited were demonstrated in a "toy problem" domain, in the future it should be possible and perhaps desirable to co-evolve policy for the agents in the Emergency Department ABM as well.

There is an implicit assumption within the ED system that patients must be handled in accordance with the core assumptions of a public healthcare facility; patients cannot be ignored, denied care, or rejected from the facility. Therefore, the ErController will interpret an invalid response from the GP as a "don't-care" condition and take an action which is valid, but likely very non-optimal. For example, if the GP evolved policy indicates that a patient is to be transferred to a room that does not exist, the ErController will map the room to one that exists, although the room may be full or in use, in which case, an alternate room will have to be chosen by the nurse upon arriving to the room. Similarly, most randomly generated policies (as opposed to a policy that results in completely random behaviour) will effectively transfer all patients to the same treatment room, to which all doctors are assigned. Effectively, some patients will get treated, but the utilization of the doctors and treatment rooms is nearly worst case. In contrast, the worst possible evolved policy would result in no patients being treated. The latter is a

``

pathological case that occurs relatively rarely through accumulated random changes in the evolved polices themselves.

In addition to minimizing ILI causing virus spread within the ED, constraints are placed on the wait time of medium and high priority patients. This is based on discussions with Brandon Regional Health Authority staff [21], although because this is an initial attempt at ED policy evolution, the maximum allowable wait times for medium and high priority patients were doubled to 3600 seconds, and 1800 seconds respectively. A penalty will be applied to the fitness of each evolved policy for each medium and high priority case that violates this constraint.

It is important to as much as possible provide an "even playing field" for the varying agent policies being evaluated. Otherwise, in this case, one would be selecting for policies that experience favourable initial conditions. This is true of the ED ABM system as evidenced by the strong correlation of ILI causing virus spread with initial ABM conditions in Chapter 6. Therefore, a controlled and separate random number stream is used for the patient generators, in order for identical patients to be generated for each policy for a particular fitness evaluation. An open source implementation of the Mersenne Twister [22] pseudo-random number generator of the with c++ API [23] is employed. If the default c random number generator were used instead, a "bifurcation-like" phenomenon would take place, where each behaviourally distinct policy would take different actions at some time t, resulting in a different *number* of calls to the default random number generator, resulting in a different random number generator state at time t+1, resulting in different patients arriving on or after t+1.

``

Furthermore, the evolved policy is only concerned with the manner in which patients are handled after the triage process is complete. A static policy, virtually identical to the one used in Chapter 6 is used to handle patients before triage is complete. Any ILI causing virus spread between patients and staff before triage is completed is not under the control of the policy and serves as an additional random variable (between individual policies) with respect to initial conditions. Therefore, to again, "level the playing field" all ILI causing virus spread for patients before the triage process is complete, is suppressed (does not occur).

The new version of the ABM for APO within the ED assumes that once triaged, patients will proceed individually to one of two waiting rooms, and will proceed individually to the exit once their treatment is complete. All other transfers within the ED, between treatment rooms, and waiting rooms require the transfer nurse to accompany the patient. The ErController will only permit the GP generated policy to request a transfer only if there are no pending transfers ordered by doctors, or transfers required to fulfil initial room assignments once patients have completed triage. Specifically, there is a transfer queue within the ErController which stores pending transfers (as patient, destination room tuples). The ErController will only enqueue a transfer on behalf of the evolved GP policy if and only if the transfer queue is empty.

As with a real ED, a policy to manage infection spread will need, as input, the condition of each patient (infected or not). This is accomplished by flagging patients as having a clinical impression of ILI as part of the triage process. The accuracy of this assessment is modeled as being 90% effective. Therefore, some patients may be treated as ILI positive when they are not infected, and others may be viewed by the policy as

``

uninfected when they are ILI positive. The addition of Partially Observable Markov Decision Process (POMDP) aspects to the APO task, is reasonable and reflective of reality with respect to challenges faced within an actual ED.

7.3.3 ABM Parameter Choices

Figure 7.14 illustrates not only the appearance of the new transfer nurse agent type, but also that the floorplan for the ED layout has been changed in order to offer more options for the automatically generated policy to quarantine or cohort infected agents. This hypothetical layout is based loosely on consolidated observations of the Brandon Hospital ED, St. Boniface ED, as well as the Health Sciences Center ED in Winnipeg, Canada. The layout features a staff lounge where the nurse and doctor agents begin the simulation at time t=0. There are two waiting rooms with a capacity of 30 patients each, and 11 treatment rooms which can be used as temporary waiting rooms for up to four patients. As mentioned in section 7.3.1 all patients besides the patient being treated must be transferred elsewhere before the doctor can actually begin treatment. Because the APO task is not concerned with ED processes before triage, the number of registration and triage nurses (desks) is fixed at one each.

Chairs which can harbour ILI causing virus and serve as transmission vectors, were discussed in Chapter 6, but not enabled for that study. For APO in this chapter, "infectable" chairs (ones that are capable of harbouring the ILI causing virus) are enabled to further penalize policies that promiscuously cohort ILI positive patients and susceptible patients together. An infectious patient has a probability of "infecting" (causing the chair to harbour the ILI causing virus) the chair they are sitting in, equal to

``

the close transmission rate found in Table 1 of section 6.1.3, that is, 0.0002 per tick as long as the patient remains sitting in the chair. Susceptible patients that sit in a chair which is harbouring the ILI causing virus have a probability of being infected equal to 0.00005 per tick as long as they remain seated in the chair. "Infected" chairs harbour the ILI causing virus for 10800 seconds or ticks, at which point the chair will no longer be able to infect susceptible patients that sit there. One chair is present in each treatment room and 12 chairs are present in each of the waiting rooms. It is assumed that an available chair will be used by a patient 100% of the time while in the treatment room. Similarly, agents in waiting rooms will sit in available chairs. If all chairs in a room are occupied additional patients stand until a chair becomes available.

``

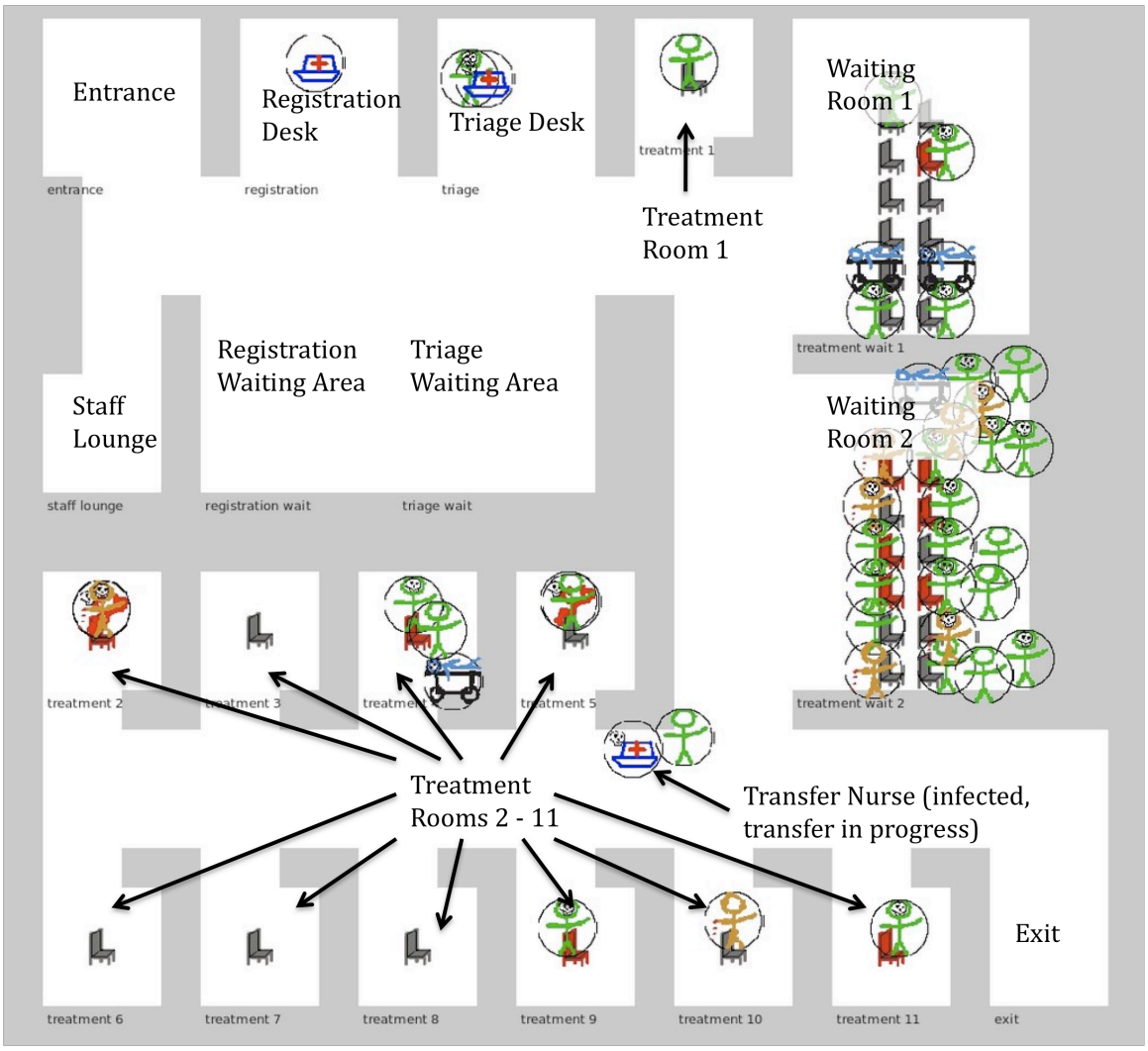

Figure 7.14. Floorplan layout of the ED used in the APO task.

Patient arrival rates for the three patient classes are the same as in Chapter 6. However, to represent a ILI surge-like scenario, 40% of arriving low priority patients are infected with ILI causing virus, whereas for simplicity all healthcare workers (HCW) are initially *not* infected. The immunity rates of patients and HCW are 0 for this simulation, since it is intended for the automatically generated policy to manage ILI causing virus

``

spread rather than some other policy like an immunization policy. Although it has been previously discussed, attrition of low priority patients is not simulated for the APO task.

7.3.4 On Hallway Medicine

One pervasive aspect of ED care in Manitoba is what is referred to colloquially as "hallway medicine" usually referring to ED overcrowding as manifest by numerous patients lining ED halls, typically on gurneys. From discussions with various healthcare professionals, administrators, and IT staff [24] it is clear that hallway patients have typically completed their treatment at the ED and are waiting to be transferred to a hospital ward, or to another hospital. They are waiting in the ED hallway because there is no available bed in their destination ward. Since they have completed their process through the ED, these patients could equally well wait in the hallway of their destination ward, and could be treated as having exited the ED. Whether they are forced to wait in the hallway of the ED, or in the hallway of their destination ward is strictly a matter of hospital policy, and therefore the latter case is chosen for simulation, or rather *not* to simulate hallway medicine in the APO task.

7.4 Genetic Programming Extensions

As stated in section 7.1, the motivation for this work is to solve a difficult problem, presumed to have an underlying MDP or POMDP in a style similar to Reinforcement Learning (RL). Extensive work has been done using RL as an approach to solve MDP and POMDP class problems [25][26]. As with RL, the solution is unknown at the outset, but the quality of the policy can be measured and used as a feedback or reward signal [1].

``

As with RL, the policy takes an action given some state of the environment, potentially changing the state of the environment, and receives a reward. Unlike a typical RL algorithm, in the case of GP, the rewards are accumulated throughout a training episode and used as the fitness score of the individual GP policy.

To this end, this section proposes a heuristic which introduces the notion of "current state" to GP. Other researchers in the field have suggested that this heuristic extends the representational power of standard Genetic Programming. Section 2.5 discussed in detail the challenges of choosing how to partition the problem space in various ML approaches, especially when considerable expert domain knowledge and analysis is required in domains with continuous environment state variables, or a large number of discrete values. Laborious expert involvement in tuning these ML parameters can lead to a relatively low "A to I ratio"[6]. The remainder of this section will propose and explain a heuristic which is in accordance with the principle of "high A to I ratio" whereby the GP evolves the number of "states" to use, as well as the rules to transition between these "states" where each state refers to an evolved GP expression tree. In fact, if the concept of state is not useful for the task at hand, the same heuristic can easily be used by GP to represent things such as decision trees, looping, and other constructs. Therefore, this again moves towards freeing the GP designer from becoming an expert in each application domain. In this work, the heuristic presented is called MASH for MArkov State Heuristic.

Markov Decision Problems underlie many real world problems, most notably many problems involving decision making under uncertainty. Managing ILI causing virus spread within a simulated ED, with unknown future arrivals, as well as uncertain ILI

``

diagnosis, clearly falls under this broad category of problems. When using a Reinforcement Learning algorithm such as Q-Learning the choice of state variables, sensor or other inputs, affects the number of states available for the Reinforcement Learning algorithm to make use of. If a wide variety of inputs are used, then learning quickly becomes computationally intractable. If too few inputs are provided then the result is too few states to represent the problem domain sufficiently to solve the problem.

The discussion here is in the spirit of Plausible Reasoning [27], this discussion will demonstrate that MASH is a useful heuristic in a complex domain, adding credibility to the conclusion that MASH is a useful heuristic in general. Koza himself, declines to make any proofs in his seminal book, Genetic Programming [14] instead spending 800 pages or so providing evidence, adding to the credibility that Genetic Programming itself is a valuable Machine Learning technique. Aside from the motivation provided earlier in this section, the discussion relates to MASH in general, and not specifically with respect to the APO task.

7.4.1 Fundamentals of MASH

Previously, each GP individual consisted of only one expression tree. In the case of GP with MASH, each individual solution is now comprised of one GP expression tree for each of the individual's states. Figure 7.15 shows two individuals, one with three states created and one with two, out of a maximum of 10 possible states. Individual 1 has 0,3, and 7. Individual 2 has 0 and 5. The details of the individual GP trees for each state are not shown here, but the GP trees are of the same type discussed in section 7.1. Whenever the GP-Individual is executed, a specific indexed memory location is interpreted as an

``

integer to determine which state's GP expression tree to begin executing. By convention, this memory location has the index 0 – this chapter will sometimes refer to it as next_state. This memory persists from one time the individual is executed to the next, and can be equally accessed by all states comprising the same individual. Since indexed memory, including next_state, can be altered during the course of executing the GP expression tree, this could lead to potentially very complex behavior.

It is necessary to ensure that the state stored in next_state is a valid index, given the maximum number of allowable states. Invalid values of next_state are mapped to valid ones. If the indicated next_state is valid, but is not yet created within the individual, it is created before it is executed. In such cases, the new state is constructed from existing states taken from individuals within the population. In this case, MASH will first try to copy the corresponding state from another GP individual, starting with the best fitness individuals. If the to-be-created state does not exist in any other individual, a random created state in the individual being executed is chosen and copied as the new state. The aforementioned functionality for masking errors, and creating states on demand, was developed during early experimentation with the MASH technique. Formerly, new states were only created when they were requested between generations, in all other cases, the execution of the non-existent state was masked by mapping it to an existing state. There were two problems with this: new states were very rarely created, and when they were, the resulting individual wouldn't be evaluated until the *next* generation resulting in poor performance.

In addition to being created upon request, there are also state-level variation operators of single-point crossover, n-point crossover, insertion, and deletion which are applied

``

between generations immediately after the regular variation meta-operators are applied (eg. regular tree crossover). A single-point state-crossover event between two individuals is shown in Figure 7.15 with the effect of exchanging the complete GP expression trees of uniformly selected states as functional chunks. Note the states that are chosen for crossover do not need to have the same index. The n-point (state) crossover is conservative in the sense that crossover of identical parents will result in identical offspring to the parents. During an n-point crossover event, if a corresponding state exists in both parents then there is a 50% chance of the expression trees in those states being swapped between parents. That is, in the example in Figure 7.15 only state 0 has a 50% chance of being swapped between parents. Insertion copies one state from the donor to the recipient, often increasing the number of states in the recipient by one. Finally, unused or rarely used states can be culled by the application of the deletion operator, with a weighted probability to bias deletion towards rarely used states. State based crossover (single-point or n-point with equal probability) is applied 80% of the time when individuals are replicated, and state based insertion or deletion are performed the remaining 20% of the time.

In a process distinct from the state deletion variation meta-operator explained in the previous paragraph, a separate process is applied before individuals are selected for replication into the next generation; for all the states in all individuals, if the state usage count (the number of times the states was executed during fitness evaluation) is zero, there is a 10% chance of deleting that state.

It should be noted that when non-MASH “normal” variation meta-operations are used (e.g. crossover) that require contribution from two parent individuals, a random existing

``

state is chosen from each parent. Subtrees from each of the chosen states are then selected for exchange via crossover, for example.

The state corresponding to an index of 0 is special. This state must always exist in an individual and cannot be deleted or culled. This is because it is the default state. If the memory contents of next_state are uninitialized, execution begins in state 0 the next time the individual is executed. This may occur the first time the individual is executed during a run, although in the present implementation discussed, memory persists between runs.

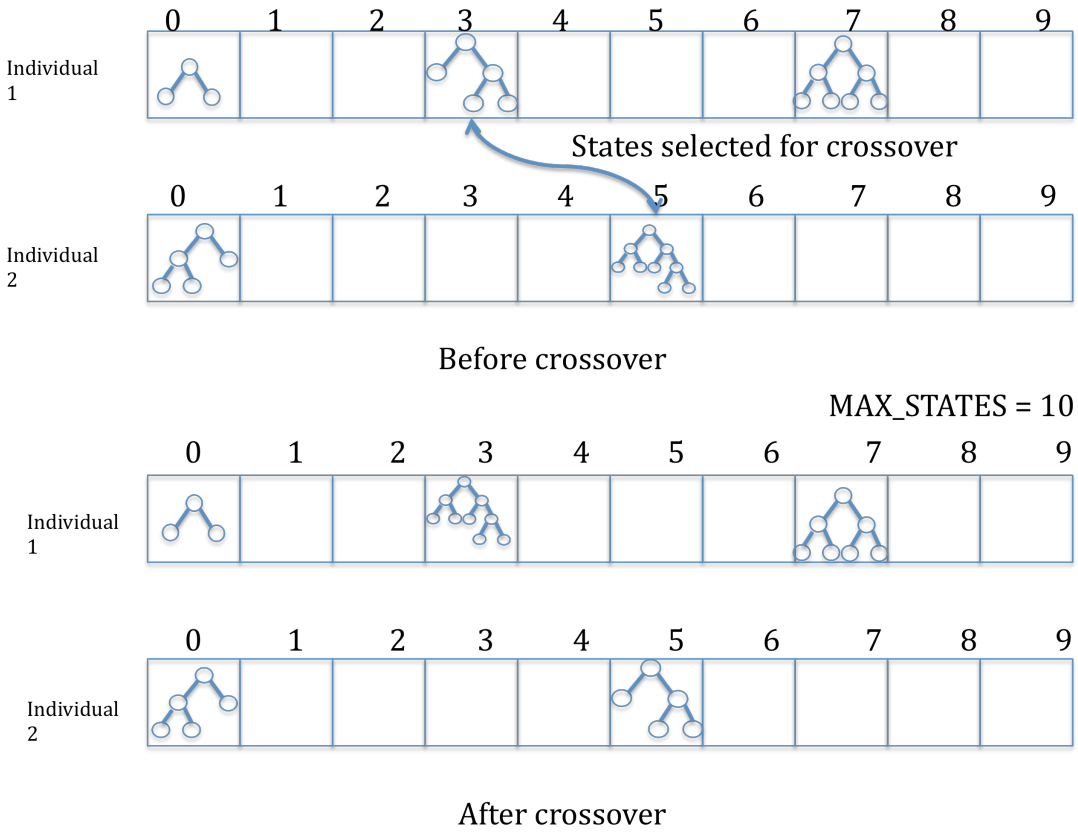

Figure 7.15. State-based crossover between two MASH Individuals

``

So far, no additional operators or terminals were required for the GP in order to implement MASH. Instead, by convention, a fixed memory location is interpreted as the next_state. By adding one more binary operator, an immediate state jump, the GP is granted use of conditional execution as well as iteration. The JUMP(X, Y) operator always returns the value of Y regardless of whether the "jump" occurs. In the case that the argument Y is nonzero, execution of the tree corresponding to the current state halts immediately; the stack is cleared, and its size set to 0.; if the state corresponding to X does not exist, then X is mapped to a valid state; then execution resumes beginning with the first terminal leaf in the tree corresponding to the new state, X. Subsequent states may also jump to further states. Execution of states will continue until there are no more jumps taken and the root node of any state is executed, at which point the function will return. Execution may also stop once some predetermined number of symbols have been executed (keeping count between states) in order to prevent infinite looping (and thus avoiding the problem of infinite loops). The following examples demonstrate conditional execution as well as iteration in the following examples, utilizing JUMP(X,Y). Additional operators for MASH are summarized in Table 7.3.

Table 7.3. Additional Operators Required to Implement MASH

Symbol	Operator	Description
?	JUMP(X,Y)	IF Y is nonzero, execution of the tree belonging to the current state halts immediately. The stack is cleared, and its size set to 0. If the state corresponding to X does not exist, then X is mapped to a valid state. Then, execution resumes beginning with the first terminal leaf in the tree corresponding to the new state, X. Returns Y.

``

7.4.2 MASH Examples

One of the primary motivations stated was interest in problems with an underlying MDP or POMDP. Therefore, the discussion will demonstrate how the Markov Model in Figure 7.16 can be implemented using MASH.

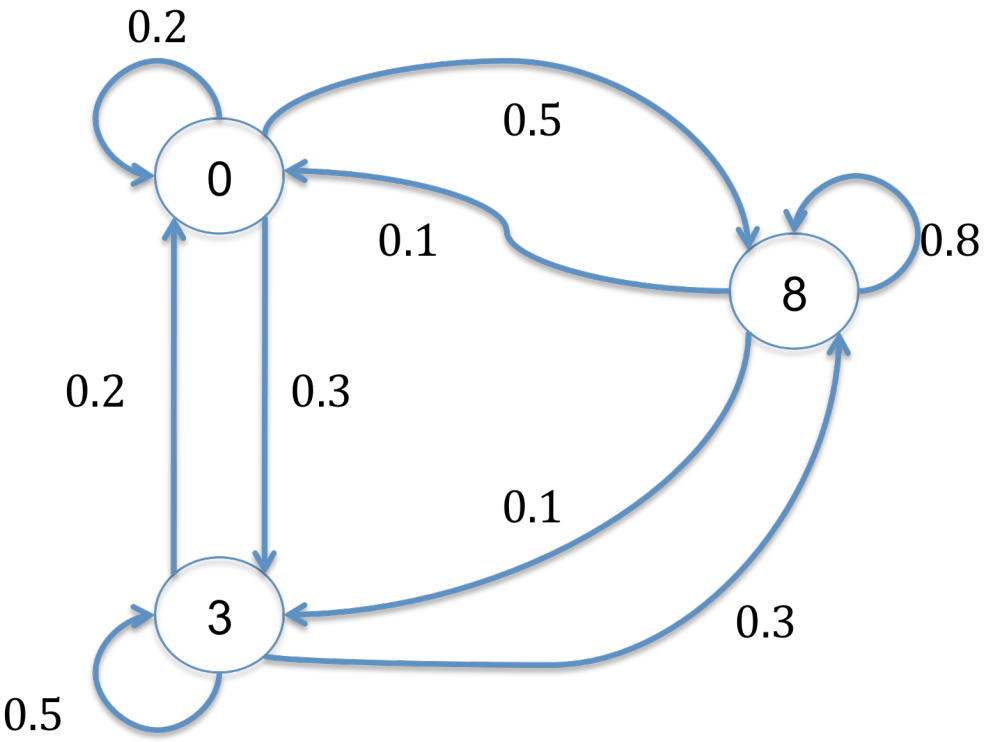

Figure 7.16. Arbitrary Markov Model

The GP-tree for state 0 is shown in Figure 7.18. Note that the first argument to operators is the left branch, and the second argument is the right branch. Pseudocode corresponding the GP-tree for state 0 is found in Figure 7.17.

``

```
1   ALIAS: random_float => memory[6]
2   ALIAS: next_state => memory[0]
3   next_state := 0
4   random_float := genrateRandomNumber()
5   IF(random_float > 0.2 AND random_float < 0.7) THEN next_state := NULL
6   IF(next_state IS NULL) THEN next_state := 8
7   IF(random_float > 0.7) next_state := NULL
8   IF(next_state IS NULL) THEN next_state := 3
```

Figure 7.17. Pseudocode corresponding to the GP-tree (Figure 7.18) for state 0 of the Markov Model (Figure 7.16)

It is fairly straightforward to demonstrate that that MASH can be used to represent states 3 and 8. One need only replace the constants used as state indices and replace the constants derived from state-transition probabilities. Although the individual solution in Figure 7.18 may appear inelegant, this is only one of many possible programs that have the same result. The style of program that the GP evolves will likely be radically different from the style of human programmers. Consider the driving force, fitness based selection versus (hopefully) conscious or intelligent experiential best effort design.

``

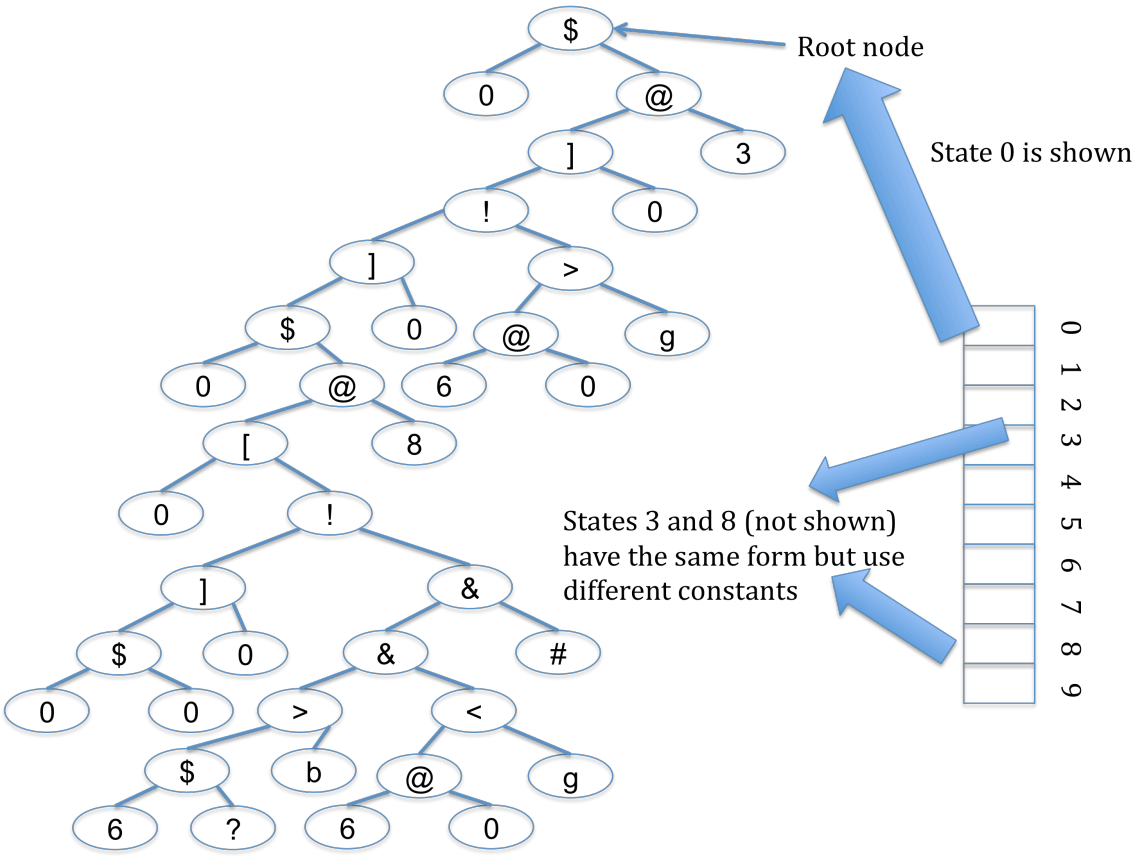

Figure 7.18. MASH representation of the Markov Model (Figure 7.16)

The earlier discussion mentioned that adding the JUMP operator to MASH is sufficient to enable looping and conditional execution. Figure 7.19 shows a flow chart for an arbitrary IF-ELSE construct, and Figure 7.20 shows the MASH implementation of that IF-ELSE construct.

``

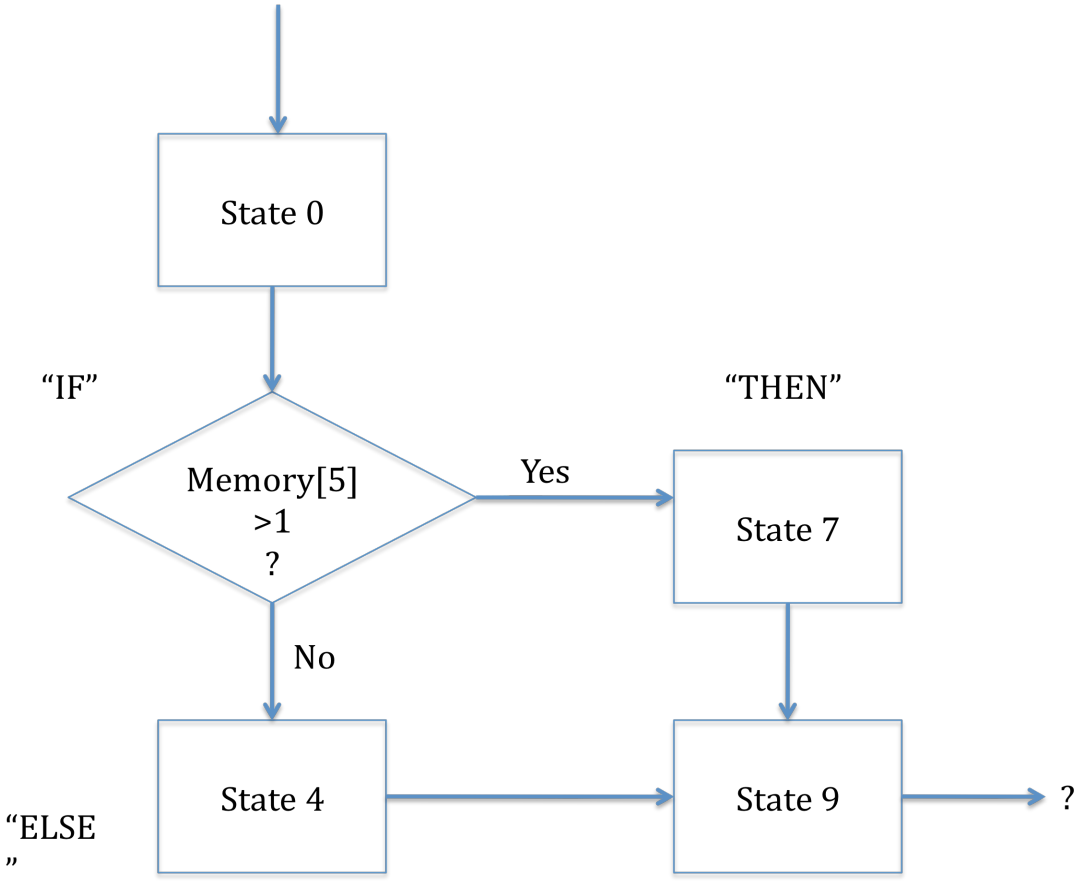

Figure 7.19. Flow chart for IF-ELSE construct

``

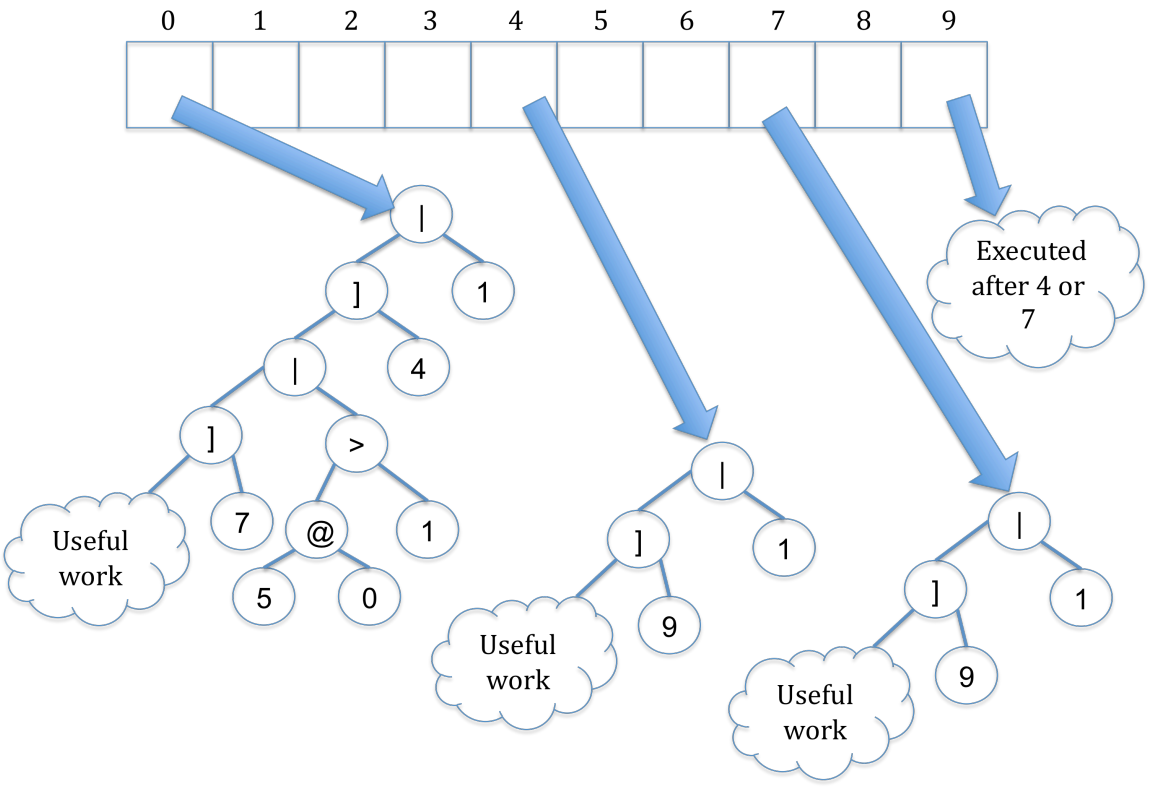

Figure 7.20. MASH implementation of IF-ELSE construct

Figure 7.20 shows the individual has four states. Beginning with state 0, there is an opportunity for the GP to perform some calculation or action before the conditional jump. If the condition is true, then state 7 is executed. Otherwise state 4 is executed. Both 7 and 4 jump to state 9, where the program flow rejoins.

Finally, the MASH implementation of the loop expressed as a diagram in Figure 7.21 is shown in Figure 7.22, and pseudocode can be found in Figure 7.23. In this example the individual has 3 states, 0, 3, and 6. State 0 is responsible for initializing the loop variable stored in memory with index 5, then jumps to state 3 where the condition check and body of the loop are contained. State 3 immediately checks to see if the loop condition has failed. If so, it jumps to state 6. Otherwise it performs whatever work it is supposed to,

``

updates the loop variable by subtracting 1, then jumping to state 3 again, this time with a different loop variable stored in memory with index 5.

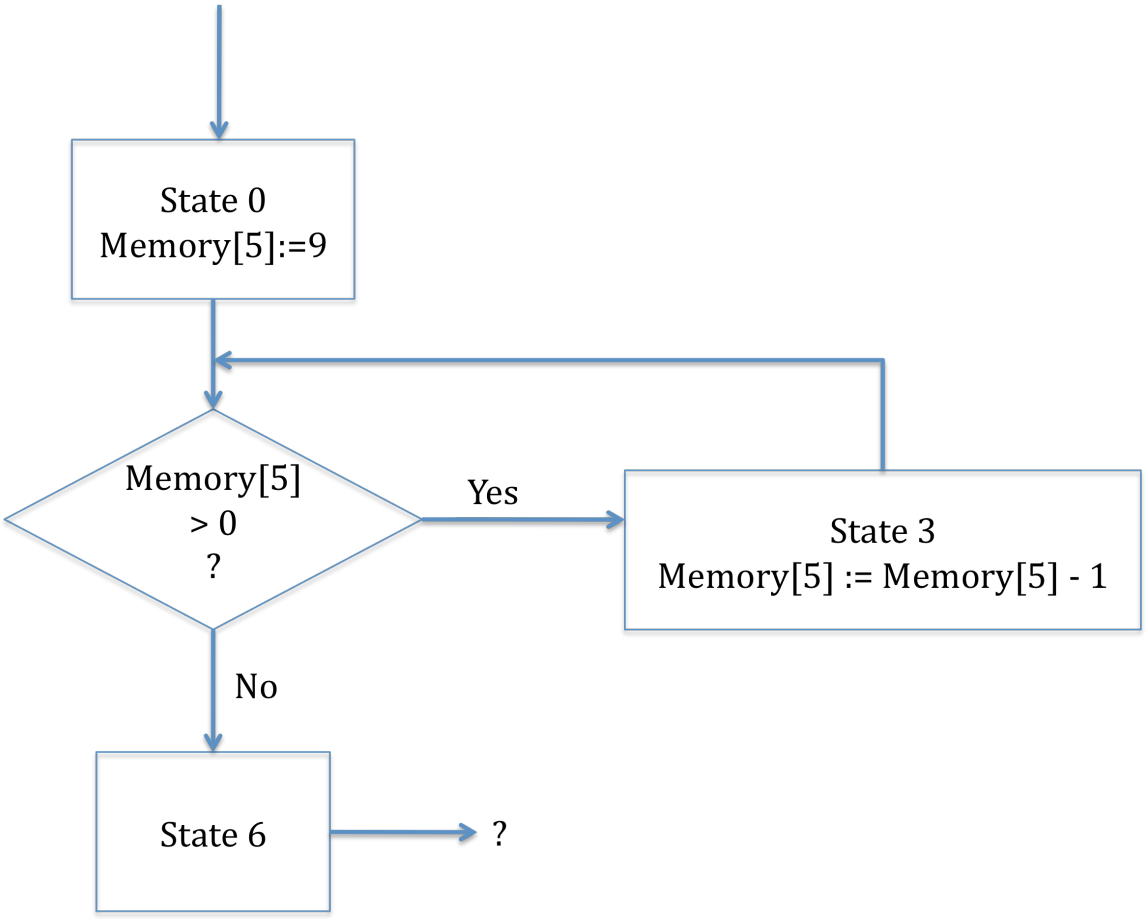

Figure 7.21. Diagram for arbitrary loop construct.

``

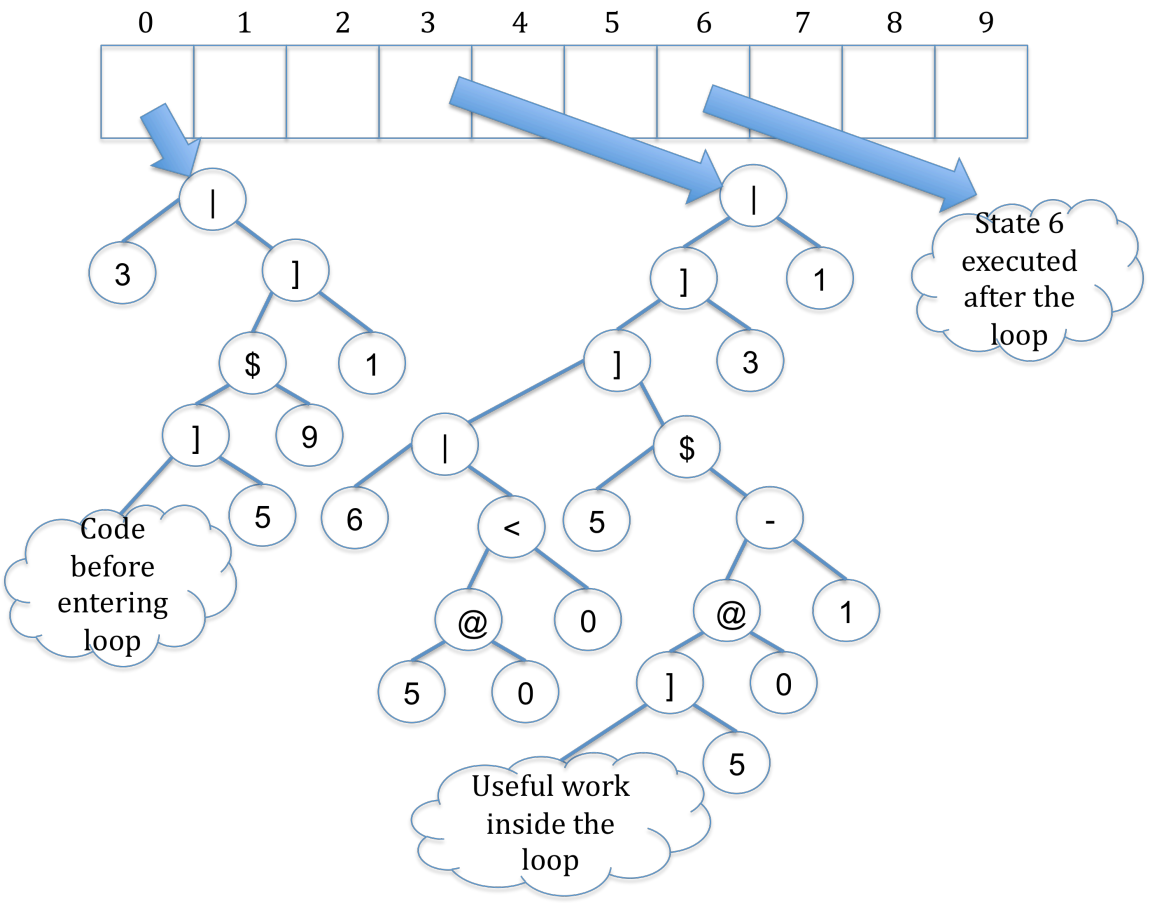

Figure 7.22. MASH tree for implementing the loop construct.

```
1. Memory[5] := 9
2. WHILE(Memory[5] > 0){
3.    perform repeated useful action
4.    Memory[5] := Memory[5] - 1}
5. perform actions after the loop
```

Figure 7.23. Pseudocode for loop construct.

``

7.4.3 Related Work

Although MASH is conceptually unique in the way that it introduces current state to GP, it does share features with multiple existing GP implementation variants. One obvious similarity to Linear Genetic Programming [28] is the use of indexed memory or registers for input from, and output to the environment.

It is interesting to note the similarities and differences between MASH and Koza's Automatically Defined Functions (ADFs) [14]. ADFs are GP subroutines that co-evolve along with the programs that use them. While both approaches appear to promote modularization and reuse of code, inspiration for ADF came from function or method calls in programming, whereas inspiration for MASH came from current state in MDPs. ADFs build on Koza's standard tree based GP written in Lisp, much like MASH builds upon this C++ GP implementation. Koza's implementation is quite rigidly structured, with special nodes, and one side or branch of the tree reserved for ADF definitions. The same "main program" branch of the singular expression tree is executed each time; in contrast, the structure of a MASH individual includes a separate GP expression tree for each state, one or more of which will be executed depending on the current_state, and evolved state transition rules.

One basic semantic difference is notable. A subroutine maintains the state of the calling routine, and therefore once the subroutine is complete execution resumes where it left off in the calling routine. A change of state is different in that there is no expectation to return execution to the exact spot where it left off, so the stack of the calling routine is not maintained. Aside from the JUMP(X,Y) operator, state changes occur between instances of GP execution. MASH also appears to have an increased design emphasis on

``

versatility. If the state heuristic isn't useful, the same infrastructure can be used to achieve looping and conditional constructs. Also unlike ADF's and, aside from JUMP(X,Y), basic MASH doesn't add any terminals or operators, it again reuses a facility that was already present in the GP system.

MASH can emulate many of the key features of ADFs. That is, the representational capacity is present to make use of these features. They may arise spontaneously, but the system doesn't explicitly select for solutions making use of these features.

The first feature of ADFs [14] is the capability to decompose a problem into subproblems. MASH can clearly represent this, as each state could potentially solve part of a problem. Koza claims that ADFs are capable of recursive application of the decomposition process. MASH does not have this ability, although procedural decomposition may be possible in some cases using loops. ADFs feature identical reuse of solved sub-problems. MASH too is capable of this, by virtue that it can repeatedly cycle through certain states that contain solutions to a sub-problem. MASH, like ADFs can perform parameterized reuse of solved sub-problems. This is possible if MASH uses shared memory to implicitly pass arguments back and forth. MASH does not have any provisions for Abstraction, unfortunately – as there is no provision to reduce the variety of available symbols to any one state. This is not necessarily detrimental, as GP has demonstrated some capability to cope with extraneous terminals and operators [14].

Similar to the nodes that ADFs need to protect in order for the system to work, MASH reserves a special status for the state with index 0. State 0 is analogous to some of the structural nodes in ADFs and cannot be deleted. It is the default state, and care must be taken to ensure it exists. In MASH, however, there are no immutable tree nodes.

``

In both systems decomposing the problem into subproblems may promote human readability since the human may be left with smaller, easier to deal with chunks. The more recent extensions that permit states to be created when needed, the constraint that state 0 must exist could likely be relaxed, but this has not been tested. Koza's ADF approach introduces a "structure preserving crossover" operator in order to prevent alteration of the structural nodes, while MASH introduces its own state-level variation meta-operators, described in 7.4.1.

There are GP implementations described where GP evolves state transition rules for Cellular Automata whose structure was arrived at by some other means [29]. There are also instances where a finite state machine (FSM) evolves "symbiotically" with a GP expression tree at each state. GP-Automata [30] and later FMS(GP) [31] share similar goals with MASH, of generating modular and reusable code, ultimately adding credibility to the MASH approach. FSM(GP) is an extension of GP-Automata which allows for a variable number of states in each individual, as well as adding several new types of mutation. FSM(GP) and GP-Automata don't use indexed memory, instead have separate structures which co-evolve state transitions explicitly, rather than MASH implicit encoding of an FSM or Markov Model. As a result, the GP cannot change the initial state during execution, rather only by mutation, compared to MASH which can alter the initial state during the course of execution by writing to memory index 0. Furthermore, since state is explicit, and strongly coupled with the function of the GP system, the use of states in GP-Automata and FSM(GP) is not optional. MASH allows for greater versatility, evolving FSM-type structures only if it improves the fitness of individuals. The previous section demonstrated that MASH-GP could potentially encode other program structures

``

such as decision trees in such cases where the FSM encoding is not useful. Finally, Recurrent Tree Networks [32] introduces another way to provides state and state transitions to GP, however this approach does not include any persistent memory, indexed or otherwise, instead electing to pass state through arguments during the current execution cycle. Therefore, execution begins with the same state every time the GP individual is executed, and does not maintain a "current state" between execution cycles.

7.5 The Combined ABM-ML System for Agent Policy Optimization

In order to perform the Agent Policy Optimization (by means of evolution) task described in section 7.2, the extensions to the ABM described in section 7.3 were made. The Machine Learning system used to perform the APO task is described in sections 7.1 and 7.4. This section is specific to the APO task, summarizing the configuration of the ED ABM, the MASH enhanced GP, and provides a few additional details related to the APO task itself. As mentioned earlier, the general configuration of this system is that the ABM acts as the environment in which the GP generated policies are evaluated, and their fitness recorded.

As previously discussed, the ErController must administer queues, and room assignments in order to minimize infection spread or maximize patient flow. Although it could be loosely said that this is a multiobjective optimization problem, in reality it is likely that these two are confluent goals. The ErController is an agent whose job is roughly analogous to a clinician, or electronic decision support system whose job it is to

``

direct patient care. An alternative mode of operation would likely evolve individual agent's policies separately. As mentioned, previous work by other authors [18][19] has co-evolved cooperative behaviour in agents. Thus, an alternative ABM-ML system may co-evolve separate doctor and nurse policies. Figure 7.24 illustrates that the ErController acts as an interface and wrapper for the GP to interact with the doctor, transfer nurse, and patient classes. For more details on the individual state transitions for these classes, please see section 7.3. Note that the GP's 0^{th} memory element is reserved for storing the next state for MASH, so the 1^{st} memory element is used to pass the requested room ID from the GP to the ErController and the 2^{nd} memory element is used to pass the requested priority ID to the ErController.

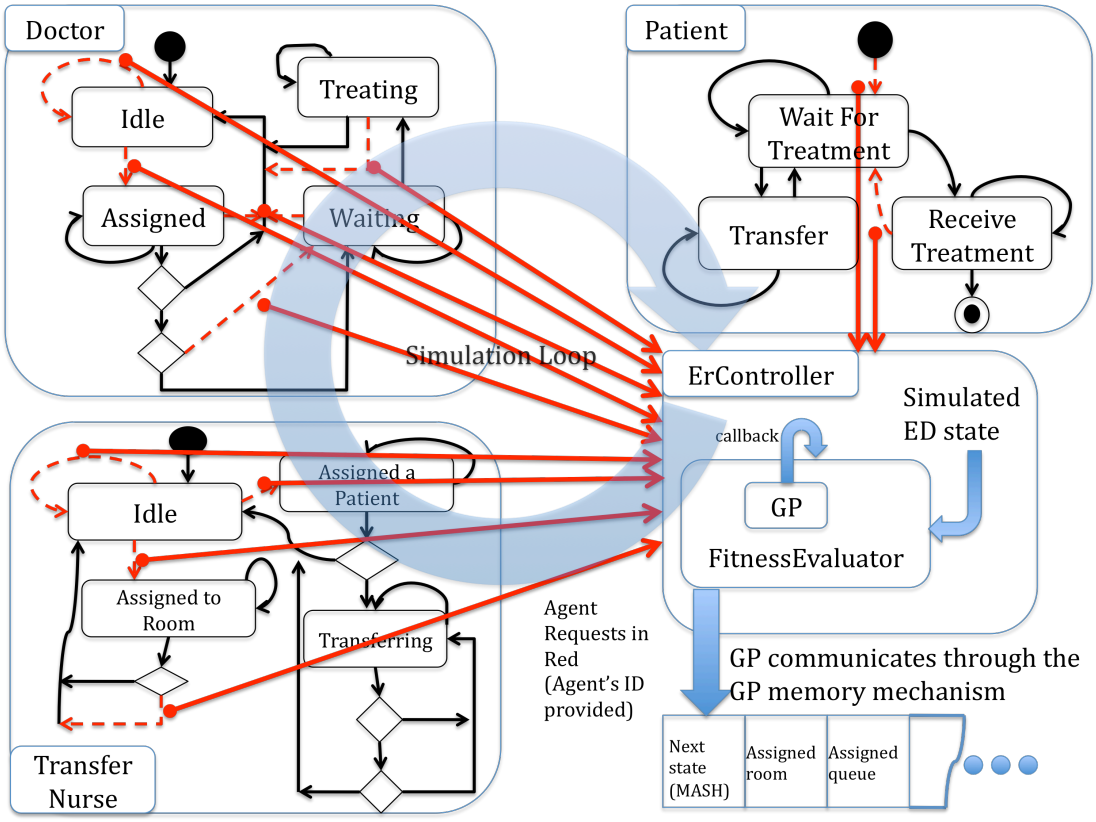

Figure 7.24. General flow of combined ABM-ML during fitness evaluation.

``

7.5.1 Implementation of the Combined System

To increase the versatility and reusability of the core Genetic Programming and MASH computer code (written in C++), the GP system was decomposed into several components or classes designed to reduce coupling between core GP classes and application specific classes. Effort was taken to ensure that the system is comparatively application agnostic, that is, any application specific code is kept separate from the core GP classes; EvolutionEngine, Population, and, GP_Individual.

The EvolutionEngine contains all the code necessary to transform a generation after fitness evaluation into the next generation. It contains methods or functions to sort the population so that rank selection can be performed, followed by replication and application of the variation operations such as standard crossover and mutation, as well as the state level variation operators defined by MASH. This class also contains the methods needed to initialize a random population at the start of a GP run. The Population class simply acts as a container for all the GP_Individual instances that make up the population. The GP_Individual itself is merely a container for the genome(s), and associated indexed memory which are merely arrays of characters, and floating point numbers, respectively. Figure 7.25 captures the relationships between all the GP classes once fitness evaluation is complete and the next generation is being constructed.

``

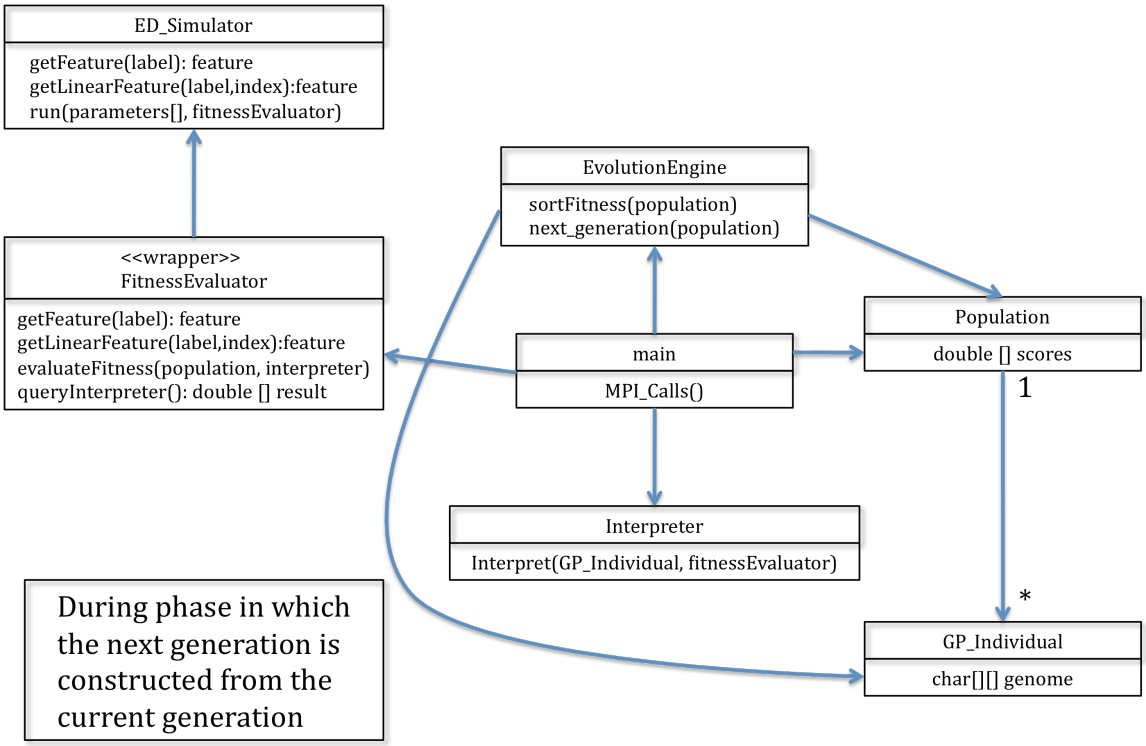

Figure 7.25. Instance Diagram of ABM-ML System while Constructing the Next Generation

Application specific classes include the FitnessEvaluator and Interpreter. The FitnessEvaluator is an interface to and wrapper for the application-specific fitness evaluation of each GP_Individual. In this case the fitness is evaluated by the ABM. When the agents in the ABM query the GP for the appropriate action to take, the Interpreter processes the particular GP_Individual's genome as well as the current state of the ABM. The Interpreter returns some output to the ABM, again using the FitnessEvaluator interface, and the FitnessEvaluator measures the quality of the GP_Individual by the result of the decision within the ABM. The relationships between the ABM-ML system classes during fitness evaluation are shown in Figure 7.26. A class diagram is not shown, as all GP components are non-derived, or base classes.

``

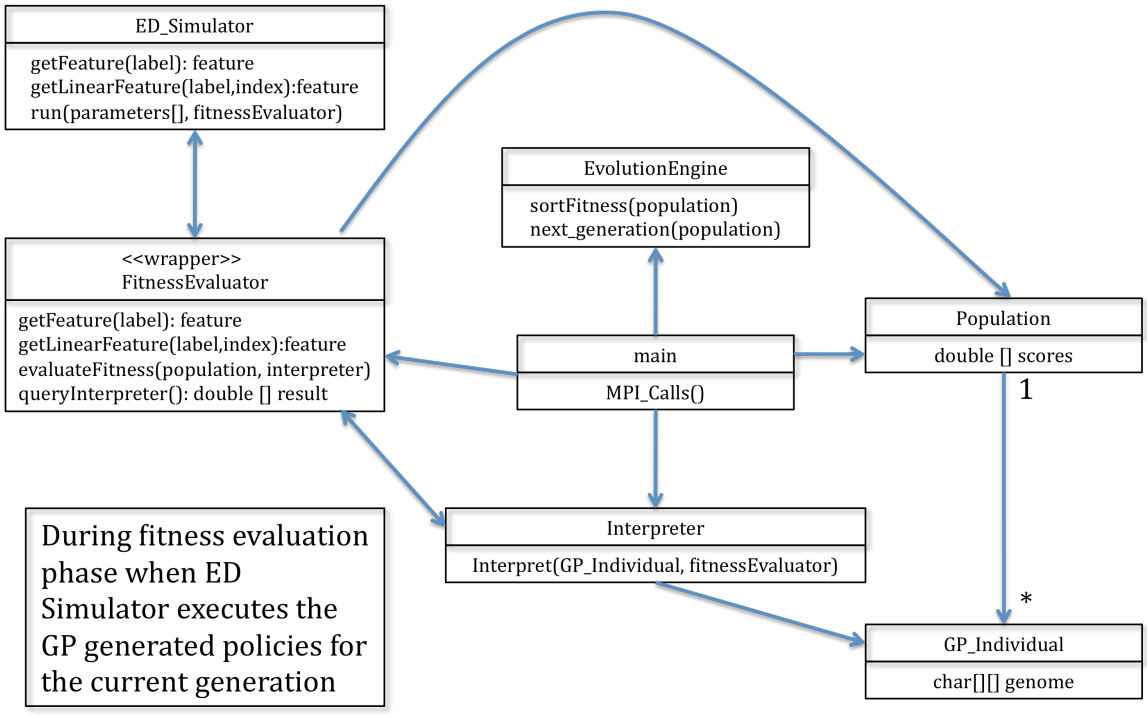

Figure 7.26. Instance Diagram of ABM-ML System while Evaluating Fitness

7.5.2 Genetic Programming with MASH implementation details

The implementation details listed here are based on the resources available for the run, as discussed in section 7.7. In each generation, the fitness of each GP generated policy is determined by evaluating the performance of the policy within the ABM on 10 different random seeds. That is, in each generation, each policy is tested 10 times on different random seeds resulting in different incoming patient flows encountered by the GP_Individual encoding the policy. Regarding the specific implementation of the ABM (ErController) and FitnessEvaluator interface, the GP returns its output, O_i , by means of writing to indexed memory. Using MASH, the value of the memory at index 0 is interpreted as the next MASH state. The value of the memory at index 1 is then

``

interpreted by the ErController as an integer indicating the desired room ID, and the value of the memory at index 2 is interpreted by the ErController as an integer indicating the desired priority or queue ID. Each ABM instance makes 20 queues available for use by the GP. In contrast, typically the output of traditional GP is the last symbol remaining on the stack when the GP-tree has been completely traversed is used as the return value. Before delving into the application specific terminals and operators used to approach the APO task, the system level parameters which control the operation of MASH will be discussed. The specific parameters discussed for the remainder of this section are regarding a run of the combined ABM-ML system, the results of which are presented in section 7.7.

The specific fitness function used to evaluate and rank each individual in the population takes into account effectiveness at managing patient flow, timely treatment of medium and high priority patients, and mitigation of ILI causing virus spread. Recalling that if all rooms of a given type are full, the simulation terminates early, the FitnessEvaluator records what percentage of the run was incomplete. The FitnessEvaluator also records what percentage of patients not infected with ILI causing virus became infected during the course of the simulation. During testing of the implementation it became apparent that the GP appeared to be working to terminate the simulation early by overcrowding the waiting rooms. This would in turn prevent ILI causing virus spread by drastically cutting the simulation time in which ILI causing virus has in which to spread. To prevent this degenerate outcome, when a simulation terminates early, all uninfected patients in the simulation are counted as having acquired the infection. Also the patient generators are run as if the simulation continued until the

``

natural end time, and the number of uninfected patients that would have been generated are counted. These patients too are counted as having acquired the infection. Thus, terminating the simulation early results in the worst possible outcome. In section 7.3, maximum allowable waiting times for medium and high priority patients were defined. The FitnessEvaluator counts any such patients that wait more than the allowable time. Again, to penalize any individuals that cause the simulation to terminate early, all medium and high priority patients in the ED, as well as any such patients that would have been generated during the simulation are counted as having waited longer than allowable. These factors combine such that for the APO task fitness is defined as:

Raw fitness = -1 * ((*percentage of entering patients that acquired infection during the ED visit*) + (*percentage of simulation time remaining after early termination*) + (*percentage of medium and high priority patients that waited longer than allowable*))

Then,

Standardized fitness = 3 – *Raw Fitness.*

However, since selection is rank based, individuals were simply sorted by raw fitness ordered such that higher raw fitness individuals have a higher rank.

The specific population size for the run discussed in section 7.7 is 500 individuals. When creating the next generation, the EvolutionEngine will sort the individuals in order of fitness and then apply a rank based selection procedure for selecting the individuals which will be replicated into the next generation. The probability of choosing a more favourably ranked individual over another, acts as a parameter which tunes the learning rate of the system. This parameter is called $P_{selection}$ and its usage is explained in Figure 7.27. Generally, a higher value of $P_{selection}$ results in individuals with a higher fitness

``

being selected more often. Figure 7.28 shows how many times each ranked individual can expect to be selected for replication in 10,000 selection events.

```
GIVEN: List of N individuals sorted by fitness rank (best first)
INITIALIZE: candidate = first individual in list; finished = FALSE
WHILE candidate NOT past end of list AND finished is FALSE DO:
    r = random number between 0 and 1
    IF(r < P_selection) THEN finished = TRUE
    ELSE candidate = next individual in the list
END WHILE
IF candidate past end of list THEN candidate = first individual in list
candidate is the individual selected for replication
```

Figure 7.27. Pseudocode for selecting an individual for replication into the next generation.

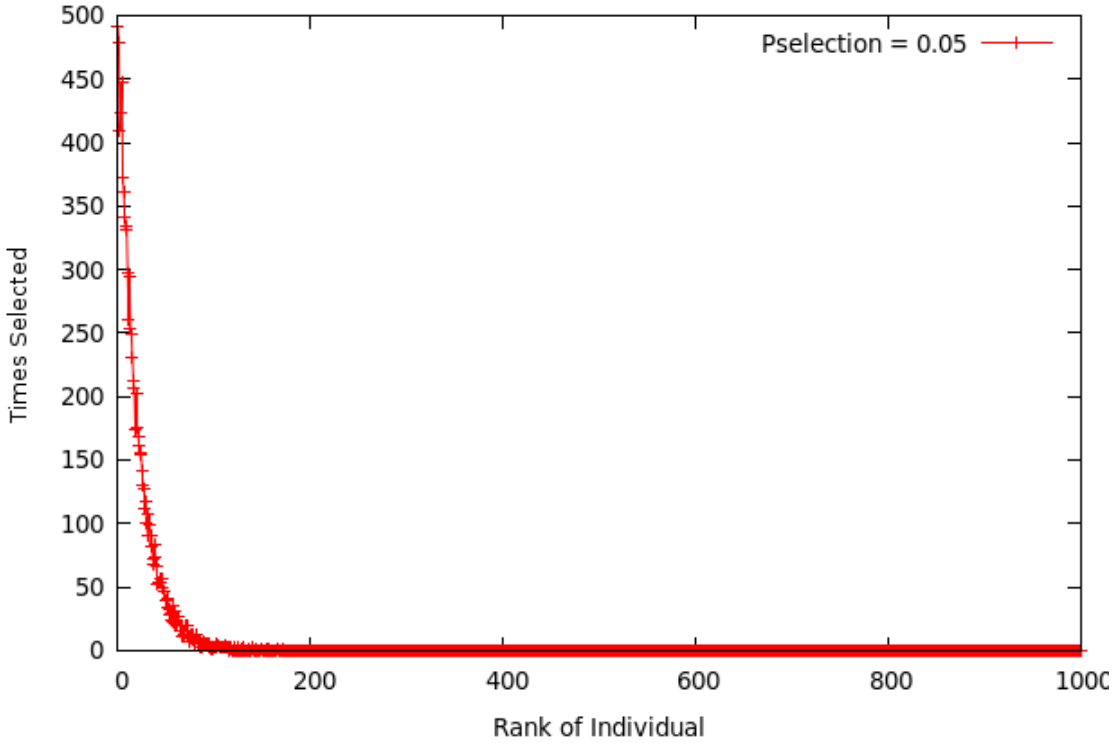

Figure 7.28. Plot of selection frequency for each rank, using 10,000 samples.

The GP system uses a form of elitism [14], where some number of top performing individuals by rank in each generation are copied without changes or chance of deletion

``

into the next generation. The number of elites used (or individuals which cannot be deleted between generations) was 20. The rest of the next generation is filled based on individuals selected as in Figure 7.27, and subsequently have the variation meta-operators mentioned in section 7.1 applied on the individuals selected for replication. The mutation rate was chosen such that most of the time one regular variation meta-operator was applied, and 90% of the time this was a crossover event, and the other 10% of the time it was one of the mutation types. Similarly, most of the time, one state-based meta-operator was applied, and 90% of these were state-crossover operations. For more details, see section 7.4.

Because MASH is able to encode loops, infinite loops are possible, so MASH–GP avoids infinite loops by setting a limit on the number of symbols which can be processed per call to the Interpreter. The count of symbols processed is not reset when the state changes (the JUMP operator is used, see section 7.4). If this limit is reached, the Interpreter simply returns control to the calling method, and the relevant memory addresses can be normally interpreted as the GP output. The symbol limit used was 10,000.

The maximum number of MASH states was 20, and each state could contain up to 100 symbols. At the end of a generation, after fitness evaluation and sorting, states which were never executed during the course of fitness evaluation are probabilistically culled 10% of the time. Later attempts to execute the now non-existent state will result in the error being masked, and execution continues with a state that exists. This takes place whenever any invalid state is specified as the next state. The GP indexed memory size was 20 floating point elements. A summary of this discussion is presented in standard

``

GP Tableau format, in Table 7.4, below. A discussion detailing application specific terminals and operators follows.

Table 7.4. Tableau for the Agent Policy Optimization (Evolution) task.

Objective:	Find an agent coordination and control policy which minimizes the spread of Influenza Like Illness and maximizes patient flow within a simulated Emergency Department.
Terminal Set:	0,1,2,3,4,5,6,7,8,9, a, b, c, d, e, f, g, h, i, j, k, l, m, n
Function (Operator) Set:	“, #, $, %, &, ‘, (,), *, +, -, /, <, =, >, ?, @, A, B, C, D, E, F, G, H, I, J, K, L, M, N
Fitness Cases	10 random patient flows or seeds per generation
Raw Fitness:	-1 * ((percentage of entering patients that acquired infection during the ED visit) + (percentage of simulation time remaining after early termination) + (percentage of medium and high priority patients that waited longer than allowable))
Standardized Fitness:	3 – raw fitness
Hits:	N/A
Wrapper:	ErController
Parameters:	M=500, G=40
Success Predicate:	None
Selection Criterion:	Rank Based
Elitisim:	(Top) 20 Elitist individuals, that cannot be altered (by crossover and mutation) or deleted
Other options:	MASH, max. 20 states, max. 100 symbols per state, max. 10,000 symbols read before halting

For the Agent Policy Optimization task, the input into the GP from the ABM, V_t , can be accessed through application-specific Terminals and Operators. Terminals include the unique ID of the agent currently querying the ErController for direction; the specific type of agent currently querying the ErController (constant integer labels representing patient, nurse and doctor); the number of doctors on duty; the current number of idle doctors; the current number of patients under the care of the ErController; the current number of patients with a CTAS score that would be considered high; the current number of patients

``

with a CTAS score that would be considered low; the number of patients in the ED that have yet to pass through triage; the number of treatment rooms; the number of waiting rooms; and the current length of the transfer queue. These application specific terminals are summarized in Table 7.5. The complete set of terminals used for the run discussed in section 7.7 is the combination of Table 7.5 below and Table 7.1 which appears in section 7.1.

Table 7.5. Application Specific Terminals for APO task

Symbol	Terminal Name	Description
d	current_agent_id	unique ID of the agent currently querying the ErController for direction
e	current_agent_type	specific type of agent currently querying the ErController
f	num_doctors	count of doctors on duty
g	num_idle_doctors	current count of idle doctors
h	num_patients	current count of patients under the care of the ErController
i	num_low_CTAS	current count of patients with a CTAS score that would be considered low
j	num_high_CTAS	current count of patients with a CTAS score that would be considered high
k	pre_triage_patients	count of patients in the ED that have yet to pass through triage
l	num_treatment_rooms	count of the treatment rooms in the ED
m	num_waiting_rooms	count of the waiting rooms in the ED
n	transfer_queue_length	current length of the transfer queue

Turning to application specific operators; “getPatientIdByQueue(queue, position)” will return the ID of a patient in the specified position in the specified queue; “getAgentId(agent_type, i)” will return the ID of the i-th agent with the specified agent_type (patient, nurse, doctor); “getPatientIdByCTAS(C, i)” will return the ID of the i-th patient triaged into urgency category C (low, medium, high); “countCTASinRoom(id, C)” will return the number of patient with CTAS category C (low, medium, high) occupying the room with the specified ID; “countCTASinQueue(id, C)” will return the count of the patients with CTAS category C (low, medium, high) in the queue with the specified ID; “orderTransfer(patient_id, new_room_id)” will, if possible, add to the transfer queue that the patient with the specified ID is to be

``

transferred to the room with the specified ID, and returns a value indicating whether or not the transfer was successfully enqueued; “changePriority(patient_id, queue_id)” will reassign the patient with the specified ID to the queue with the specified ID, and return whether or not this operation was successful; “pullDoctor(id, queue)” will cause the doctor with the specified ID to turn idle, the patient they were treating to be assigned to the specified queue, and will return whether or not this operation was successful; “getPatientIdByRoom(room_id, i)” will return the i-th patient occupying the room with the specified id; “getPatientByRoomAndCTAS(id, C)” will return the patient occupying the room with the specified ID, and with CTAS category closest to C that has been waiting the longest; and “getDoctorByRoom(room_id, i)” will return the ID of the i-th doctor in the room with the specified ID.

The operator, “getPatientFeatureById(id, label)” will return the specified feature of the patient with the specified ID. Features which can be queried by label are: whether or not the patient is presenting with ILI, the ID of the doctor currently assigned to the specified patient, the patient’s CTAS score (low, medium, high), the patient’s currently assigned queue, the patient’s currently assigned room, and how long the patient has been waiting.

The operator, “getDoctorFeatureById(id, label)” will return the specified feature of the doctor with the specified ID. Features which can be queried by label are: whether or not the doctor has ILI symptoms, the ID of the patient the doctor is currently treating, the CTAS category of the patient being treated, and the ID of the doctor’s currently assigned room.

``

The operator “getRoomFeatureById(id, label)” will return the specified feature of the room with the specified ID. Features which can be queried by label are: room capacity, whether or not the room is full, a count of the patients occupying the room, a count of the doctors occupying the room, an indication of whether the room is in such a state that a doctor can begin treatment of the single patient within.

The APO task specific operators discussed above are summarized in Table 7.6, below. All features involving an indicator whether or not agents are infected with the ILI causing virus are partially observable, that is with some uncertainty. 90% of patients will be classified correctly, while the remaining 10% will appear to be infected when they are uninfected and appear uninfected when they are infected. Finally, all operators need to be programmed in such a way that they must be able to handle any range of arguments gracefully. All operators used for the APO task are included in Tables 7.2, 7.3, and 7.

``

Table 7.6. Application Specific Operators for APO

Symbol	Operator	Description
A	getPatientIdByQueue(queue, position)	return a the ID of a patient in the specified position in the specified queue
B	getAgentId(agent_type, i)	return the ID of the i-th agent with the specified agent_type (patient, nurse, doctor)
C	getPatientIdByCTAS(C, i)	return the ID of the i-th patient triaged into urgency category C (low, medium, high)
D	getPatientFeatureById(id, label)	return the specified feature of the room with the specified ID
E	getDoctorFeatureById(id, label)	return the specified feature of the doctor with the specified ID
F	countCTASinRoom(id, C)	return the number of patient with CTAS category C (low, medium, high) occupying the room with the specified ID
G	getRoomFeatureById(id, label)	return the specified feature of the room with the specified ID
H	countCTASinQueue(id, C)	return the count of the patients with CTAS category C (low, medium, high) in the queue with the specified ID
I	orderTransfer(patient_id, new_room_id)	if possible, add to the transfer queue that the patient with the specified ID is to be transferred to the room with the specified ID, and returns a value indicating whether or not the transfer was successfully enqueued
J	changePriority(patient_id, queue_id)	reassign the patient with the specified ID to the queue with the specified ID, and return whether or not this operation was successful
K	pullDoctor(id, queue)	cause the doctor with the specified ID to turn idle, the patient they were treating to be assigned to the specified queue, and will return whether or not this operation was successful
L	getPatientIdByRoom(room_id, i)	return the i-th patient occupying the room with the specified id
M	getPatientByRoomAndCTAS(id, C)	return the patient occupying the room with the specified ID, and with CTAS category closest to C that has been waiting the longest
N	getDoctorByRoom(room_id, i)	return the ID of the i-th doctor in the room with the specified ID

7.6 Parallelism

For the Agent Policy Optimization task, fitness evaluations of different GP individuals are independent, therefore a prime candidate for exploitation of computational parallelism. Genetic Programming is well known for its scalability with respect to parallelism [6].

The code base for the ABM-ML system discussed is C++, which makes possible use of the extremely popular library for parallel scientific computing, MPI (Message Passing Interface) [33]. To perform parallel fitness evaluation, the MPI process (each running on

``

a separate core) with rank=0 broadcasts the current generation as well as a series of random seeds to worker processes. The number of worker processes is the number of processors allotted to the GP run (by the computer cluster resource allocation policy) minus one (the head node). The head process will then assign work to the worker processes. Each worker process has its own instance of the Interpreter and FitnessEvaluator, which in turn wraps the worker process's own ABM instance. These classes allow each worker process to perform a fitness evaluation. Each unique fitness evaluation consists of a particular GP_Individual to evaluate as well as a random seed to use for the fitness evaluation. Once the particular ABM simulation has completed, the worker processes simply return the calculated fitness of their assigned GP_Individual, and the head process collects them. If there are still any fitness evaluations to be performed, the head process will assign any idle worker processes more work. Once all fitness evaluations for a particular generation are complete, the EvolutionEngine that creates the next generation runs on the head process. An instance diagram for parallel fitness evaluation is found in Figure 7.29 illustrating that the GP code is decoupled from the simulator instances running in parallel as only one instance of the EvolutionEngine object is required.

``

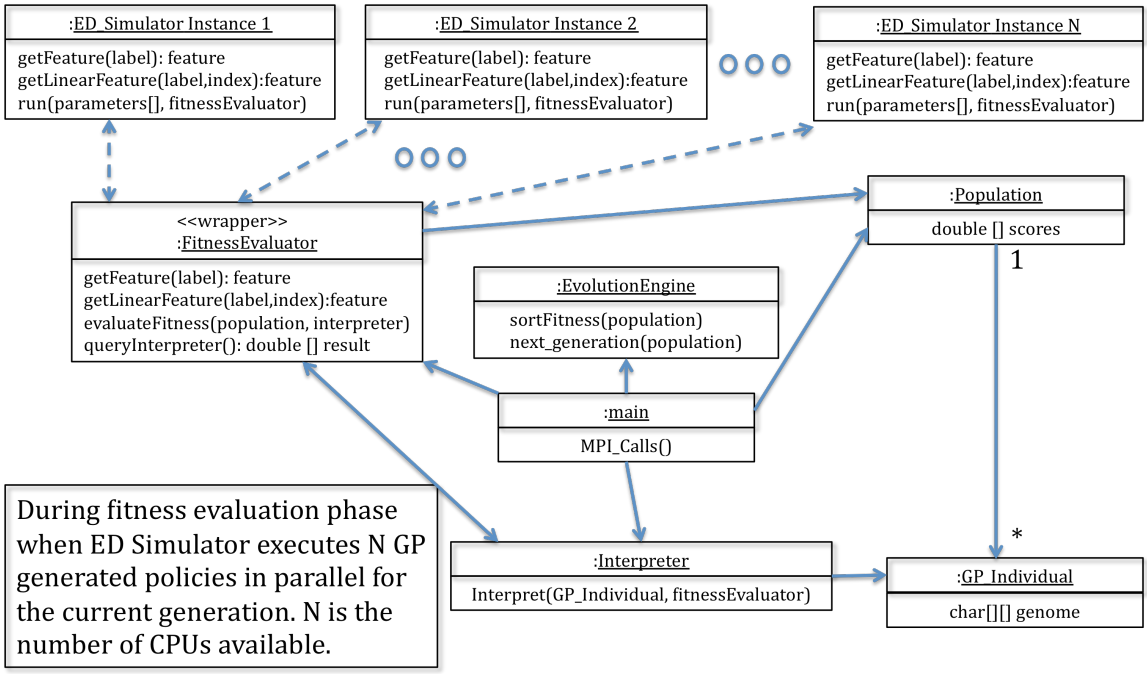

Figure 7.29. Instance Diagram for Parallel Fitness Evaluation

Implementation of parallel fitness evaluation using MPI is further facilitated by the GP_Individual's underlying implementation with simple character, integer, and floating point arrays. Broadcasting and receiving c-style arrays is relatively convenient within the MPI framework. Otherwise, broadcasting instances of GP_Individual would require serialization of objects. In the implementation discussed here, objects are stored natively in a serialized format.

``

7.7 Experiments, Observations, and Results

Although the use of Qt4 within the ABM makes implementation of many features extremely convenient, overall the use of Qt4 has presented both opportunities and challenges. At the outset of this work, it was anticipated that the necessary compute time would be available on the WestGrid compute grid for Canadian researchers. At the time of this writing, however, WestGrid is unable to provide a cluster with an up-to-date Qt4 environment. Subsequently, Dr. Vladimir Okhmatovski generously donated the use of 160 CPUs on the University of Manitoba ECE department's compute cluster. After several false-starts, compounding the difficulty in acquiring compute resources, a run of 40 generations was completed using the combined ABM-ML system to perform the APO task. As mentioned in section 7.2, it is desirable to evaluate evolved solutions using input that the GP has not yet encountered.

The results below reflect the performance of the MASH-GP system on a validation set of random-seeds, that is, input (patient flow) that was not seen during training. Yet, to maintain consistency over the validation set results over the course of generations, the same validation set of 10 random seeds is used to evaluate each generation. For comparison, an unchanging strategy where every program branch is taken stochastically is employed, designated "random" in Figure 7.30, it is the constant horizontal line, as no learning occurs over the course of generations. In addition to the random assignments of doctors and patients to rooms, this stochastic strategy will interrupt a random doctor 5% of the time when a new patient arrives, and assign a random transfer (random patient to ran-

``

dom room) 5% of the time when the transfer nurse is idle. The results of this comparison are shown in Figure 7.30, below.

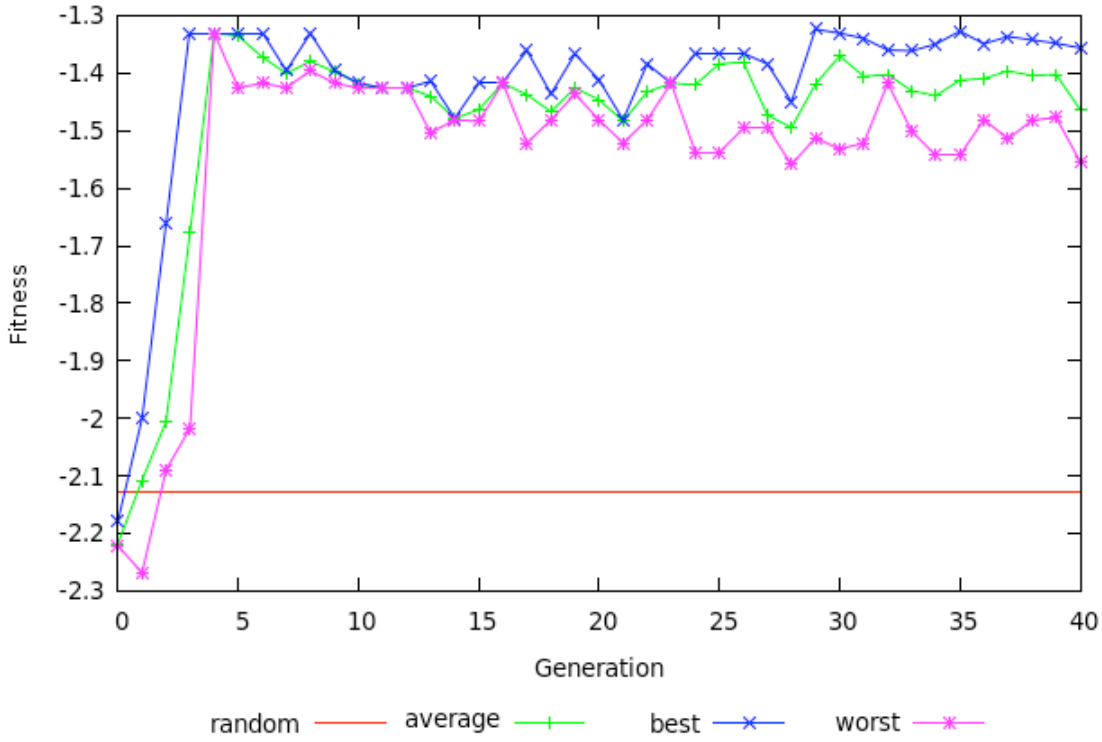

Figure 7.30. Performance of Average, Best and Worst performing individuals of each generation over the course of 40 generations compared to an unchanging stochastic strategy.

There is an improvement of the best performing individual in each generation, going from a fitness of approximately -2.2 to a fitness of approximately -1.3 over 40 generations. With increased access to compute resources, continued improvement in performance is expected, as the parallel computing approach used should scale well up to a number of CPUs equal to the population size, at least. Before going into analyzing the

``

semantics of the specific evolved solutions, we believe other steps may be of more immediate interest as outlined in section 7.8.2.

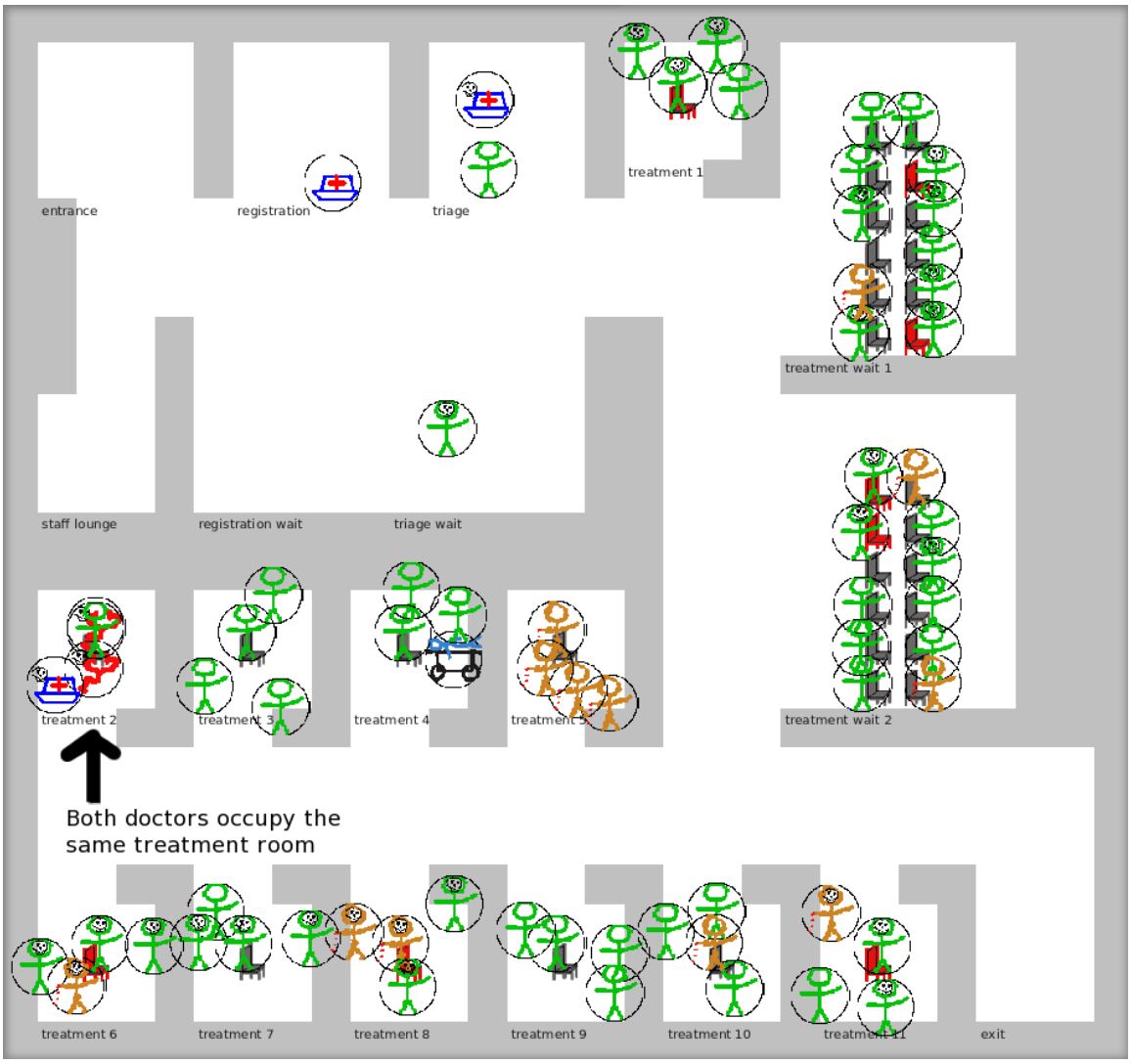

Figure 7.31. Snapshot of simulation executing best policy of generation 1.

Besides the quantitative improvement, perhaps of equal importance are qualitative observations which may be made. For instance, Figure 7.31 illustrates one particular moment of a simulation run, the best individual in generation 1, that is the best of the randomly generated initial GP population. As indicated by the figure, throughout most of

``

the simulation run both doctors spent most of their time in the same treatment room. Doctor utilization is low because only one patient is permitted to be treated in a room at a time which can be considered a social constraint based on patient expectation of privacy.

In the more evolved policy, the best individual of generation 40, produced a policy which could be considered counter-intuitive to what is considered normal practice in an ED. Figure 7.32 snapshot of the policy from generation 40 being simulated. Rather than doctors alternating between all the treatment rooms in some order, the evolved policy calls for the doctors to be stationary, with the transfer nurse using treatment rooms near the doctors as temporary waiting rooms. The transfer nurse rapidly shuttles patients from these temporarily waiting rooms to the doctor's rooms once the doctors are idle. Although this policy is likely still not optimal, this policy still represents an avenue which might not have otherwise been explored. It is also noted that the policy from generation 40 exhibited behaviour which treated high priority patients in a timely manner more frequently than the individual from generation 1.

``

Figure 7.32. Snapshot of simulation executing best policy of generation 40.

Despite the use of Qt4 causing the delays mentioned earlier, debugging was aided significantly by the built in capability for visualization made possible by the Qt4 based environment. The visual debugging of agent behaviours through observation proved to be an essential first step, prior to inspecting detailed agent state debugging information. In one particular example, this author observed a nurse start a patient transfer, and immediately afterwards, a doctor enters the room and starts treating the patient. Afterwards, the nurse proceeds alone to the destination room and waits for the patient to

``

arrive. The patient's treatment is completed and the patient leaves the simulation, resulting in the nurse waiting forever. It would have been difficult to infer this chain of events from simply inspecting the states of all the agents after the fact. Watching the agents interact, including in slow motion, can provide valuable clues as to what is causing unexpected behaviours. This is in fact a means of visualizing data in a very explicit manner.

During the run, it was demonstrated that the ABM-ML system was rather efficient on memory. The memory footprint of the GP alone is approximately 1.5MB, including both the arrays to store the population, as well as the executable code in memory. The ABM itself requires about 20MB per simulated emergency department. Both these requirements are very low for modern computers.

7.8 Conclusions

This chapter has presented a vertical slice of a decision support tool for healthcare, featuring Agent Based Modeling and simulation with a GP-based ML system. The contribution of this work has been the development of the ABM framework amenable to ML, and subsequent ABM-GP integration, Preliminary results validate the approach used, adding credibility to the statement that Machine Learning will be more routinely coupled with ABMs in the near future. By corollary, the actual evaluation and/or statistical comparison of GP-evolved ED policy remains beyond the scope of this work, however the foundations for this are clearly laid. Agent Policy Optimization, as defined, is a difficult and unprecedented ML task, presenting challenges in terms of compute

``

resources, as well as a first foray into machine augmented or directed decision support with respect to patient care, and healthcare in general.

Although preliminary in nature, at least one of the evolved policies was a counter-intuitive or unexpected result. Such policies are an example of the inventive capabilities of GP, and may open up otherwise unexplored possible policy interventions to healthcare practitioners.

This work brings socio-ethical issues to the forefront. This marks the first time that a machine evolved healthcare policy could conceivably be implemented in a real ED. If evolved policies are implemented in an actual ED, who is responsible when patients care or outcomes are compromised in the care of the ED? Issues such as these need to be more fully explored given the suggestion that countries with aging populations such as Japan will face increasing amounts of machine-directed care by automated processes or robots[34][35].

A more rigorous validation of the model and learning approach should become possible in the near future. As improved Emergency Department Information Systems (EDIS) data becomes available, the body of work for which this Ph.D. thesis is laying the foundation will validate the model. A handful of technical and political hurdles remain, including reporting errors and privacy concerns. This author does not believe that these hurdles are insurmountable, to the contrary, EDIS data accuracy continues to improve, and healthcare officials are increasingly likely to cooperate with researchers as the utility of researcher models is demonstrated, hence establishing the importance of this work.

It cannot be overstated how increased availability to HPC resources will help more rigorously evaluate MASH and the GP performance with respect to the APO task.

``

Consider the following: according to [6], GP is Mooreware, that is, able to take advantage of the exponentially increasing availability of computational power resulting from Moore's Law and the passage of time. To achieve "human-competitive results", Koza used a cluster consisting of 1000 cores, circa 2001. If Moore's law is applied very loosely, meaning there is an approximate doubling of computational power every 18 months, in the intervening nine years or so there are six doubling periods leading to an estimate of 64,000 cores today. Increases of raw CPU clock speeds of ten-fold or so mitigate this number somewhat, so a modest estimate is 6,400 cores might be an equivalently provisioned machine today. It should be noted that the Agent Based Modeling and Simulation approach is far more computationally intensive than any of the problems Koza attempts in [6], as it pushes the boundaries of today's individual machines. Much better performance is expected with better provisioned compute resources. This would also make it possible to evaluate the performance of MASH more rigorously on the Agent Policy Optimization task, since it will be possible to run a statistically significant number of runs in comparison with an alternative to MASH.

For now, this section will briefly turn attention back to Polya [27]; Plausible Reasoning can be applied to situations where there is not enough data available, or proof is intractable. One can argue that in accordance with plausible reasoning, this result adds credibility to the statement that a combined ABM-ML system featuring MASH successfully bridges the gap between available heuristics and a working solution within a healthcare policy evolution context.

Within this healthcare policy evolution context, as demonstrated in Chapter 6, the effect of chance events such as the number of arriving patients can affect the outcome of

``

the simulation to a degree comparable to the effects of simulated interventions. This level of variance only makes the already difficult problem of induction, in this case inducing an effective ED patient flow and infection spread policy, all the more difficult. Consider the following story, an excerpt reprinted from [1].

Once upon a time, there was a little girl named Emma. Emma had never eaten a banana, nor had she been on a train. One day she went for a journey from New York to Pittsburgh by train. To relieve Emma's anxiety, her mother gave her a large bag of bananas. At Emma's first bite of a banana, the train plunged into a tunnel. At the second bite, the train broke into daylight again. At the third bite, Lo! Into a tunnel; the fourth bite, La! Into daylight again. And so on all the way to Pittsburgh and to the bottom of her bags of bananas. Our bright little Emma told her grandpa: "Every odd bite of a banana makes you blind; every even bite puts things right again."

After LI and HANSON

7.8.1 Software Engineering

Throughout the course of this project, numerous observations were made regarding Software Engineering of Agent Based Simulators, the visualization provided by of such simulators, the construction of Genetic Programming systems, and the application of the latter to the former.

``

In this author's opinion, the effort invested to make good Object Oriented choices when designing the combined ABM-ML system paid off, resulting in a system that is relatively easy to maintain, as well as re-use many components for other applications. For instance, functional decomposition of the GP system into functionally cohesive classes has decoupled core GP code from application specific code to increase reusability, since application specific changes are limited to a few classes. The attention paid to reusability adds greatly to the versatility of the GP system. Take for instance the following modifications to the GP system to attempt a different task, data mining. Figure 7.33. shows that only the FitnessEvaluator and Interpreter classes need to be modified if switching target application, because application specific code is relegated to those classes.

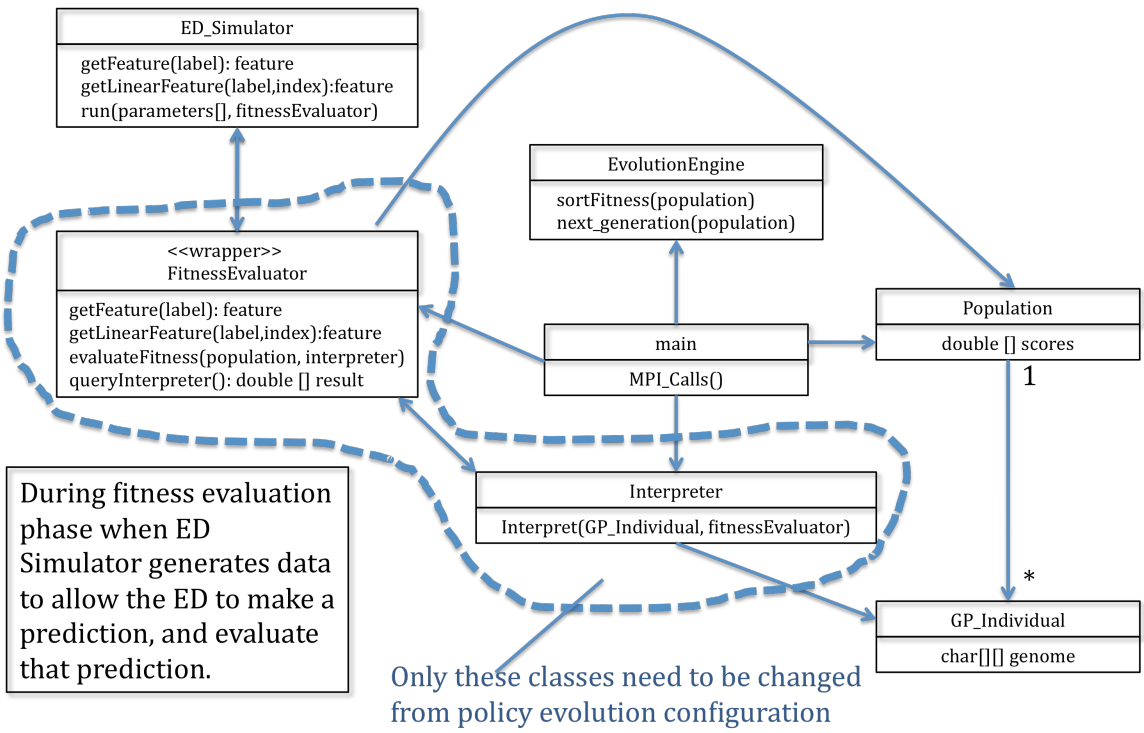

Figure 7.33. Instance Diagram for Data Mining Task, note similarity to APO task.

``

A second, minor change, more towards a traditional data mining configuration, is shown in Figure 7.34. The ABM has been replaced with a data file or database connection. Instead of querying the ABM for various ED variables, the data contained in the file or database is returned instead. In this case the FitnessEvaluator wraps the file or database interface, and the only class that needs to be modified from the example portrayed in Figure 7.33 is the FitnessEvaluator.

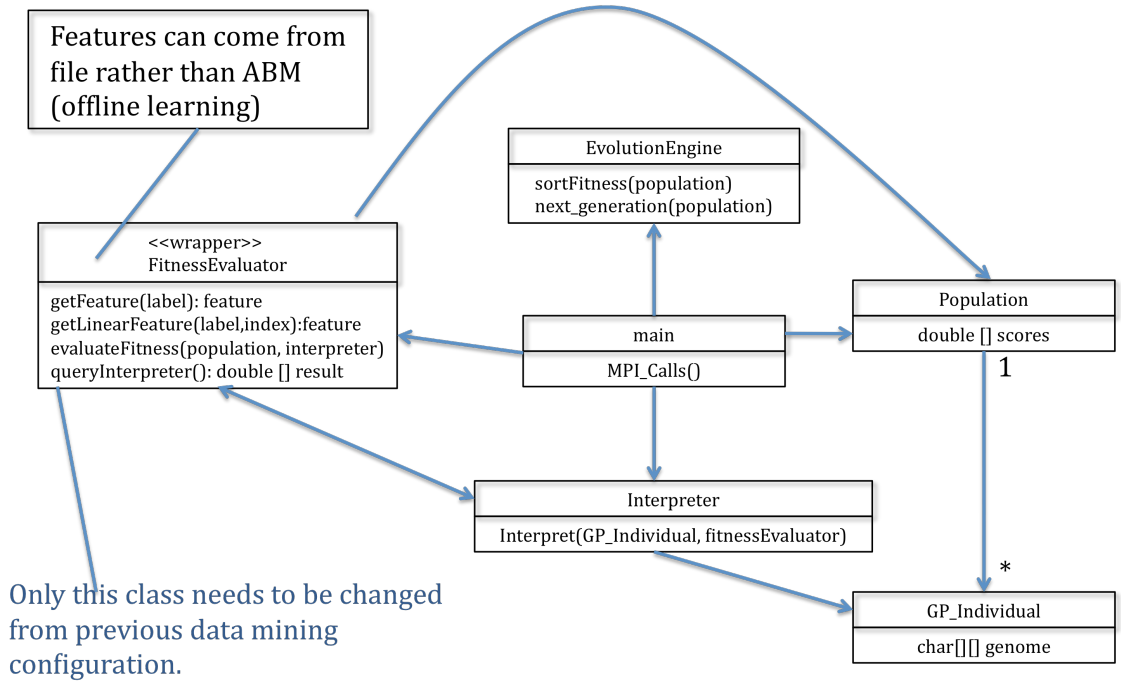

Figure 7.34. Instance Diagram for More Traditional Data Mining Configuration.

Successful parallelization of the APO task was made possible by decoupling the graphical user interface code from the core simulator code. This permits running the ABM in batch mode, without visualization, or using MPI on a compute cluster. The granularity of parallelism is individual ABM runs, each evaluating one GP individual for

``

a particular random seed. This allows for efficient load balancing, as cores are kept busy once their currently assigned simulation is completed. In comparison, the time to create the next generation on the head process is relatively insignificant, so the worker processes do not stay idle for long between generations.

With regards to Qt4, the utilization of the framework increased the speed at which the initial ABM was developed, even the core ABM code uses some convenient Qt4 classes. However, since the initial version of the ABM, the use of Qt4 has made it inconvenient to deploy the ABM on compute clusters where Qt4 is difficult to deploy, and as a result has delayed larger scale runs considerably. Although there are no inherent incompatibilities between Qt4 and MPI, due to practical considerations, it would be desirable to refactor the ABM code, separating any use of Qt4 from the core ABM code, enabling it to be deployed in a parallelized batch mode on systems where Qt4 support is lacking. Overall, the net effect of using the Qt4 framework for visualisation was positive, as visualization was invaluable for debugging the ABM extensions discussed in section 7.3. One could visually inspect agent behaviours that violated expectations of how the system ought to function.

It is important at this point to underscore the "administrative step" [6] of designing a fitness function carefully crafted to elicit the desired behaviour. Often times a means of search such as Genetic Programming will find unexpected ways to exploit a poorly crafted fitness function, evolving surprising but undesirable traits. Expertise in crafting the fitness function is the extent of the expertise required of the programmer in conjunction with the practitioner. The skill of encoding the desired behaviour or result within the fitness function has a significant domain-independent component, which the

``

programmer can build upon between applications. Regardless, the step of specifying the desired result of any Machine Learning application is common to all ML approaches.

7.8.2 Future Work

From the outset, it was known that this work or any other single effort would not solve the challenges associated with patient flows nor infection spread through hospital emergency departments. These problems are exacerbated by social dynamics not often encountered in most traditional engineering optimization problems. These problems are very likely computationally irreducible and improvements will come through the use of more powerful modeling and simulation capabilities, both algorithmic as well as in silicon. This work has established some of the first efforts in leveraging simulation and modeling with Machine Learning in the context of decision support, and lays a foundation for the extensive efforts still required. Validation of the Agent Based emergency department simulator presented here, as well as a more rigorous evaluation of MASH performance are top priorities. It is anticipated that these things will be possible in the near future, due to expected increases in compute availability for this project as well as the increasing availability of detailed data. A performance comparison of GP with MASH and without MASH is a priority, as are testing it with a larger population size, and with a rank selection parameter more skewed towards a faster learning rate. In any case, analysis of MASH generated policies should be facilitated by introducing parsimony pressure, that is for evolution to favour more concise solutions. These are likely to be more easily interpreted by humans. Finally, an important ongoing goal is to improve the ABM fidelity and realism, moving increasingly towards data-driven

``

simulation. In the future, it might be interesting to try co-evolving cooperative agent policies, rather than evolving a central coordination strategy for agents which are semi-autonomous with respect to the ErController.

With respect to the ABM code base, portability can be improved by excluding Qt4 from the core ABM code, but leaving it accessible for the workstation version with visualization. The process of refactoring might offer a convenient opportunity to investigate the possibility of employing a finer grain of parallelism with respect to running the ABM. In this proposed version, simulations with multiple regions could run each region in parallel, exchanging state information each time step. Because the ABM appears to require so little memory, a Shared Memory parallel computing environment is an option which warrants further investigation for such an implementation. The number of available CPUs will likely be the limiting factor in large simulations consisting of many regions running in parallel.

Another way to improve GP performance on this particular multi-objective problem may be to improve the fitness function by introducing a pareto ranking method [36] for combining the performance measures for multiple objectives.

If more compute resources should become available, implementation of GP "Islands" with occasional communication of genes between islands would be beneficial. Some researchers have reported superlinear speedup in such implementations [1], at the very least, one would expect to get around the problem of a pathologically bad random starting point in the search space leading to poor results. Each island would represent a new population all starting in different points in the solution search space. Furthermore, it has been shown [37] that varying the mutation rate can lead to better results. Having the

``

aforementioned islands would allow for experimentation with various mutation rates on individual islands.

7.9 References

[1] W. Bhanzhaf, P. Nordin, R. Keller, F. Francone, *Genetic Programming – An Introduction: On the automatic evolution of computer programs and its applications*. San Francisco: Morgan Kaufmann Publishers, Inc., 1998.

[2] M. Raymer, W. Punch, W. Goodman, L. Kuhn, "Genetic programming for improved data mining: An application to the biochemistry of protein interaction," in Genetic Programming 1996: Proceedings of the First Annual Conference, 1996, pp. 375-380.

[3] J. Daida, T. Bersano-Begey, S. Ross, J. Vesecky, "Computer-assisted design of image classification algorithms: Dynamic and static fitness evaluations in a scaffolded genetic programming environment," in Genetic Programming 1996: Proceedings of the First Annual Conference, 1996, pp. 279-284.

[4] J. Koza, D. Andre, F. Bennett III, M. Keane, "Use of automatically defined functions and architecture-altering operations in automated circuit synthesis using genetic programming," in Genetic Programming 1996: Proceedings of the First Annual Conference, 1996, pp. 132-149.

[5] P. Nordin, W. Banzhaf, "An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming," Adaptive Behavior, 5:107-140, 1997.

[6] J.R. Koza, et al., *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. New York, NY: Springer, 2005.

``

[7] D. Billings, “Algorithms and Assessment in Computer Poker,” Ph.D. dissertation, University of Alberta, Edmonton, AB, Canada, 2006.

[8] M. Smith, C. Feied. The emergency department as a complex system. [Online]. Available at http://necsi.org/projects/yaneer/emergencydeptcx.pdf. [Accessed: May 10, 2010].

[9] R. Sutton, A. Barto, *Reinforcement Learning: An introduction*. Cambridge: MIT Press, 1998.

[10] T. Mitchell, *Machine Learning*. Boston, MA: McGraw Hill, 1997.

[11] Yu, B. and K. Popplewell, 1994, "Metamodels in manufacturing: a review," *International Journal of Production Research*, vol. 32, 787-796.

[12] P. Hudak, “Conception, evolution, and application of functional programming languages,” *ACM Computing Surveys*, vol. 21, no. 3, pp. 359-411, September 1989.

[13] A. Teller, “Turing completeness in the language of genetic programming with indexed memory,” in Proceedings of the 1994 IEEE World Congress on Computational Intelligence, vol. 1, pp. 136-141, 1994.

[14] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[15] S. Weiss, N. Indurkhya, *Predictive Data Mining: A practical guide*. San Francisco: Morgan Kaufmann Publishers, Inc., 1998.

[16] Robert A. Kilmer, Alice E. Smith and Larry J. Shuman, "An emergency department simulation and a neural network metamodel,", *Journal of the Society for Health Systems*, vol. 5, no. 3, 1997, 63-79.

``

[17] Yeh, J-Y.; Lin, W-S.; “Using simulation technique and genetic algorithm to improve the quality care of a hospital emergency department,” *Expert Systems with Applications: An International Journal*, Vol. 32, no. 4, pp. 1073-1083, May 2007.

[18] Andre, D. and Teller, A. 1999. Evolving Team Darwin United. In *Robocup-98: Robot Soccer World Cup II* M. Asada and H. Kitano, Eds. Lecture Notes In Computer Science, vol. 1604. Springer-Verlag, London, 346-351.

[19] J. Aronsson, "Genetic Programming of Multi-agent System in the RoboCup Domain," M.S. thesis, Lund Institute of Technology, Lund, Sweden, 2003.

[20] Schmidhuber, J., Zhao, J., and Schraudolph, N. N. 1998. Reinforcement learning with self-modifying policies. In *Learning To Learn*, S. Thrun and L. Pratt, Eds. Kluwer Academic Publishers, Norwell, MA, 293-309.

[21] Meetings with Brandon Regional Health Authority staff on ‘fact finding mission’, Brandon, Manitoba, August, 2009.

[22] Matsumoto, M.; Nishimura, T. (1998). "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator". *ACM Transactions on Modeling and Computer Simulation* 8 (1): 3–30. doi:10.1145/272991.272995

[23] G. Kuenning, *Mersenne Twist Pseudorandom Number Generator Package*, 2002. [Online]. Available: http://www.cs.hmc.edu/~geoff/mtwist.html. [Accessed: May, 2010].

[24] Meetings with Winnipeg Regional Health Authority staff on ‘fact finding mission’, Winnipeg, Manitoba, February – June, 2009.

[25] Kaelbling, Leslie P.; Michael L. Littman; Andrew W. Moore (1996). "Reinforcement Learning: A Survey". *Journal of Artificial Intelligence Research* 4: 237–285.

``

[26] Hasinoff, S.W. Reinforcement learning for problems with hidden state. Technical report, University of Toronto, 2003.

[27] G. Polya, *Mathematics and Plausible Reasoning: Volume 1 Induction and Analogy in Mathematics*. Princeton University Press, 1990.

[28] M. Brameier, W. Banzhaf, *Linear Genetic Programming*, New York: Springer, 2007.

[29] Andre, D., Bennett, F. H., and Koza, J. R. 1996. Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. In *Proceedings of the First Annual Conference on Genetic Programming* (Stanford, California, July 28 - 31, 1996). Genetic And Evolutionary Computation Conference. MIT Press, Cambridge, MA, 3-11.

[30] Daniel Ashlock, GP-Automata for Dividing the Dollar, in *Genetic Programming 1997, Proceedings of the Second Annual Conference on Genetic Programming*:18-26, 1997.

[31] Karl A Benson. Evolving Finite State Machines with Embedded Genetic Programming for Automatic Target Detection within SAR Imagery. In Proceedings of the 2000 Congress on Evolutionary Computation CEC00, pages 1543-1549, La Jolla Marriott Hotel La Jolla, California, USA, 2000. IEEE Press. f

[32] Yabuki, T.; Iba, H., "Turing-complete data structure for genetic programming," in *IEEE International Conference on Systems, Man and Cybernetics, 2003.*, vol.4, no., pp. 3577-3582 vol.4, 5-8 Oct. 2003

[33] “MPI Documents” [Online]. Available: http://www.mpi-forum.org/docs/ [Accessed: May, 2010].

``

[34] M. Tutton, "Japan paves the way in robotic research," *CNN*, September 24, 2009. [Online.] Available: http://www.cnn.com/2009/HEALTH/09/23/japan.medical.technology/index.html [Accessed June, 2010].

[35] M. Kitano, "Japan looks to robots for elderly care," *theage.com.au*, July 22, 2005. [Online.] Available: http://www.theage.com.au/news/technology/japan-looks-to-robots-for-elderly-care/2005/07/21/1121539082236.html [Accessed June, 2010].

[36] C. M. Fonseca, "An Overview of Evolutionary Algorithms in Multiobjective Optimization," in *Evolutionary Computation*, MIT Press, 1995, vol.3, no.1, pp. 1-16.

[37] Piszcz, A., Terence, S., "Genetic Programming: Analysis of Optimal Mutation Rates in a Problem with Varying Difficulty," in *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference*, American Association for Artificial Intelligence, Florida, USA, pp. 451-456 (2006).

``

Chapter 8

Concluding Remarks

This chapter provides concluding remarks to the thesis, highlighting contributions made, and also discusses future work.

8.1 Summary

Chapter One defined the scope of the problem undertaken in this thesis, that is, to design a framework for spatially oriented Agent-Based Modeling in the context of a decision support tool for health policy makers. This tool provides insights and feedback when evaluating patient flow and infection spread mitigation and control strategies in the Emergency Department. Furthermore, a conjecture was made that such a tool should be augmented with Machine Learning to assist with optimization or search tasks in relation to ED patient flow and infection spread mitigation policies. This conjecture is developed throughout the rest of the thesis. The primary contribution of this thesis is the development of a novel, flexible, and tuneable agent-based modeling framework with applications to infection spread and patient flow within a hospital ED. The secondary contribution is the demonstration of the utility of Machine Learning for automatically

generating ED policy for patient flow and infection spread mitigation. This application is novel and of emerging practical importance. The tertiary contribution of this work is the prototyping of a novel heuristic for Genetic Programming, which facilitates the expression of states which are known to govern the agents which comprise the ABM.

Chapter two surveyed Multi-Agent Systems (MAS) and Agent Based Modelling and Simulation frameworks in particular, beginning with a taxonomy of MAS. Strengths and weaknesses of Agent Based Models were discussed. A comparison the features of many prominent ABM frameworks lead to a list of desirable feature set for emerging ABM frameworks such as the one discussed throughout this thesis. Chapter two concluded by surveying Machine Learning applications within MAS and choose a candidate approach which was applied in Chapter seven.

Chapter Three presented an ABM which demonstrated the feasibility of modeling complex healthcare phenomena in terms of the constituent parts of the healthcare system, as well as rules governing those parts. The parts in this instance, were agents representing healthcare workers, patients, and even inanimate objects. The complex dynamics of the system are both inherent and emergent properties within this model. This decomposition allows ABMs to capture social details to a higher degree of realism than in previous complementary models. The work also implied the potential for ABMs to eventually address "what-if" questions concerning complex healthcare systems such as those involving infection spread within the ED. A proof of concept design was presented, demonstrating that a hospital ED can be modeled effectively in such a way, and set the stage for ABM extensions to answer "what-if" questions. Being able to address "what-if" questions allows healthcare policy decision makers to test proposed

policy changes without risking patient care or significant investment of resources. As part of a plausibly reasoned argument, showing that a policy is effective in simulation adds credibility to its implementation. Chapter Three discussed the initial development of an ABM oriented to simulating hospital Emergency Departments, either individually or with multiple interacting EDs. In an investigation of patient flow, a novel model for patient redirection was introduced, inspired by well vetted Telecommunications Engineering principles. The eventual goals of the ABM framework were articulated as supporting the ongoing incorporation of real data whenever possible, improving the fidelity of the social dynamics and topographies of the ED, as well as supporting statistical processing. As such, emerging technologies for data collection, electronic medical records systems, and Real Time Location System (RTLS) based on RFID were taken into account when designing the ABM, apparent from the spatial aspect of the simulation. Specific scenarios investigated at this stage were varying ED staffing levels and patient redirection policies between EDs in the same metropolitan area, illustrating the utility of ABM within a policymaking decision process. Even at this early stage, ABMs appeared to hold significant potential roles in healthcare applications. The observation was also made that problems involving the ED have inherent complexities owing to their social nature. These problems are "messy", forcing the decision maker to assign a cost or utility to the quality of patient care. Opportunities for applying Machine Learning within the ABM were presented, such as automatic optimization of staffing levels, as well as automatic induction of a patient redirection policy. These models are stochastic and unpredictable in nature, making validation of the model difficult in the

traditional sense, if not impossible. The work presented was one of the first times that ABM was used in the context of a decision support tool for healthcare in the ED.

Chapter Four further elaborated on the integration of emerging data sources, as these will become necessary to enhance the fidelity of, and ultimately validate these ABMs. Improved quality and availability of data was discussed in relation to further developing the model's structure, agents, their interactions, numerical simulation parameters, initial conditions, as well as being of possible input directly into ML systems. Examples of such data sources included RFID-based RTLS, wireless sensor networks, and a network-enabled smartphone platform. The results of a previous attempt at "data-fusion" were presented, suggesting that it is possible to integrate the various real-world data sources discussed, even when the data are intermittently available, noisy, or incomplete to a considerable degree. The model demonstrated increased potential for the ABM to address "what-if" scenarios at this stage, but higher fidelity data, specific to the ED being modeled was required before meaningful "what-if" queries could be made in relation to a specific ED. The investigated "what-if" scenarios involved RFID reader placements for a simulated RFID-based RTLS. Various ML applications became apparent, including automated placement of RFID readers and optimizing reader coverage for a particular budget of fixed number of readers. The ABM visualization tool was shown to be an excellent communication tool between healthcare practitioners and the modeller, aiding in model construction and configuration, and validation. Healthcare experts communicate model requirements in their natural language, model parameters are discussed using a familiar lexicon, and the behaviour of the model resembles that description, leading to fewer systemic errors in the modeling process which could be

introduced through successive abstractions. Furthermore, this approach is amenable to communicating results back to the healthcare practitioners in their natural language.

Chapter Five contrasted the ABM approach with a Queuing model approach. It became clear that the strengths of ABM are simulation and subsequent visualization of transient phenomena, whereas the strength of QM is steady-state analysis. The QM is subsequently used to investigate a "what-if" scenario comparing a policy which pre-empts lower priority ED patients in order to service high priority patients in a timely manner. Qualitative results suggest that a pre-emptive policy shortens wait times for high priority patients, and increases the wait times of lower priority patients. A non-pre-emptive policy has the opposite effect. The utility of enhancing model inputs with real-world data as it becomes available was once again highlighted. Two specific ways in which ML could augment the ABM were introduced: a predictive data mining task, namely, patient wait time forecasting; and policy evolution for automatically generating a patient redirection policy between several EDs within the same region.

Chapter Six presented a refinement of the ABM which is suitable for investigating a number of "what-if" scenarios with respect to a novel ABM for infection spread within the ED. This model was featured in an extended use-case scenario which followed a hypothetical decision making process in which the ABM was used to analyze spread of an ILI causing virus within the ED. The analysis suggested a number of statistically significant variables, on which several intervention policies were based and subsequently evaluated. This use-case scenario demonstrated the utility of the ABM approach within a healthcare policy decision process. The discussion also supported the idea that machine directed search through parameter or policy space is desirable to complement work done

by a human analyst using statistical software. The development of an ABM framework to model contact-based infection spread within an ED is a novel contribution of this work. The potential capabilities are illuminated, such that any degree of specificity in agent characteristics, behaviours, and interactions can be accommodated, ultimately constrained by available compute resources and time needed to create a model of extremely high resolution and fidelity. Limitations of the approach were also discussed, including the availability of data as a limiting factor in model resolution and fidelity. The results of the analysis that was carried out suggested that health status and numbers of HCWs in an ED exert a larger influence over overall infection spread, than patient-oriented policies. The utility of the ABM was demonstrated as a decision support tool available to healthcare practitioners and policy-makers capable of qualitatively assessing the relative impact of various infection control policies. Chance effects and naturally occurring large fluctuations in prevalence can confound the effect of interventions, and no model is yet sensitive enough to quantitatively and exactly validate the full range of infection spread dynamics.

Chapter Seven presented work which consolidated a combined ABM-ML decision support tool capable of automatically generating policy within a modeled ED. An evolutionary process was the specific means used to achieve automatic policy generation. Throughout the chapter, the primary concern was the development of a framework for machine augmented decision support systems on which a body of applications can be built. A novel Machine Learning task, Agent Policy Optimization, concerned with maximizing patient flow and minimizing infection spread was presented in detail.

Genetic Programming has been shown to be capable of evolving an internal problem representation. Owing to the numerous agent and ED states present in the ABM, a case was made for the desirability of a heuristic allowing GP to evolve its own state representation for a given problem. In accordance with the previously mentioned principle of freeing the ML practitioner from imparting domain-specific knowledge to the learning process, flexibility – even flexibility to represent structures other than states – was discussed as a high priority for the heuristic. MASH was presented as a heuristic which extends the representational power of standard GP in accordance with the aforementioned design goals. MASH was compared to a number of approaches with similar design goals of providing the concept of state to GP, suggesting that MASH is novel in the sense that it takes advantage of indexed memory, and is flexible enough to have multiple uses other than just representing state.

Preliminary results, combined with the ability of GP to take advantage of ever-increasing availability of compute resources support the argument that machine generated policies developed within an ABM framework will be increasingly common in decision support tools for healthcare. However, the full power and utility of these techniques will be realised as the technologies become available to deliver accurate and detailed information about the ED, political barriers related to information access are overcome, and a reasonable amount of compute resources are provisioned. As such, the actual evaluation and statistical comparison of GP-evolved ED policy has always remained beyond the scope of this work. This work laid the foundations for these things, and represents a first foray into an unprecedented ML task involved with machine augmented or directed decision support with respect to patient care, and healthcare in general.

Furthermore, this work has brought socio-ethical issues related to liability to the forefront, if a machine directed form of healthcare negatively impacts patient outcomes. Such questions will need to be addressed, especially in the context of rapidly aging populations requiring an ever increasing amount of machine directed healthcare.

Owing to good Object Oriented Software Engineering choices, the flexibility of this framework was demonstrated by the relatively few classes which would have to be altered to attempt other ML tasks or applications. Also, because the visualization code is decoupled from the core simulator code, it is possible to run fitness evaluations in parallel on a compute cluster. However, because Qt4 convenience classes are used by the core simulator code, in practice it makes deployment of the ABM impractical or impossible on several available compute clusters which do not support Qt4. Overall the outcome of using Qt4 has been positive, since visualization has proven invaluable in debugging the ABM-ML system.

8.2 Future Work

Taking advantage of the visualization layer that Qt4 enables, and still keep Qt4 out of core simulator classes for cluster deployment, will require a significant refactoring effort. The anticipated increased accessibility of compute resources as a result of refactoring, as well as the anticipated availability of finer data, should make a statistically significant number of runs possible contributing towards validation of the ABM. The increased availability of compute resources would enable a performance comparison GP with MASH to GP without MASH as well. This would also signify a move towards more

data-driven simulation. It is also desirable to try the APO task with a larger population size, and a rank selection parameter skewed more towards a faster learning rate. Rigorous analysis of GP evolved policies is expected to be facilitated by the introduction of parsimony pressure into the fitness function used by the GP resulting in more concise, human readable, expression trees.

During the course of refactoring, opportunities to make further improvements to the ABM-ML system will arise. Utilizing a finer grain of parallelism, using spatial decomposition as a heuristic for distributing the execution of a single ABM replication across multiple cores, co-evolving policies for different agent types simultaneously, pareto-ranking for multiple objectives, introduction of Fuzzy aggregation operators, and Fuzzy states are all foreseeable possibilities and avenues of exploration. Finally, with increased availability of compute resources it should be valuable to simultaneously evolve semi-autonomous populations of individuals, with each core representing an "island" of individuals evolving mostly independently but with occasional communication between "islands". Such schemes have reportedly produced superlinear speedup, perhaps owing in part to multiple starting populations, reducing the chances of a pathologically bad random starting population.

The refactored version of the simulator presented herein should adhere to the design guidelines developed in Chapter 2 for practical ABM frameworks:

- *conveniently extendible* to modeling human-centric domains
- strike a balance between flexibility, extendibility, and specific support for human-centric domains
- extendible to interface with Machine Learning modules

- support facilities for incorporating real-time data
- support real-time visualization for communication, model validation and development
- visual tool for model environment construction or editing environments imported from data
- amenable to providing a scripting layer on top of the compiled code, and then adding visual programming (drag and drop) on top of the scripting layer
- support for distributed, parallel, or cluster computing

Appendix A: A Mobile Platform with potential application to Real-Time Patient Location Systems

Rapid Prototyping Vehicle-to-Infrastructure Applications using the Android™ Development Platform

M. Laskowski, J. Allen, M.R. Friesen, R.D. McLeod, and K. Ferens
Electrical and Computer Engineering
University of Manitoba
Winnipeg, Canada
mlaskows@ee.umanitoba.ca

***Abstract*— This paper discusses a rapid prototyping environment for an application developed relative to the USDOT Vehicle Infrastructure Integration program. The primary application represents opportunities beyond traditional vehicular telematic applications by virtue of the technologies leveraged to prototype a vehicle crash detection system, as well as an emission sensor network for vehicle fleets and a vehicle speed retarder. A central theme in the research is that emerging vehicular telematic applications need to capitalize on advanced software engineering orientations, including open-source frameworks, web services, and Service Oriented Architectures.**

***Index Terms*—Wireless Vehicular Telematics Applications, Statistical Inference, Crash Detection, Services Oriented Architecture.**

I. Introduction

This paper discusses a vehicle safety application developed relative to the USDOT Vehicle Infrastructure Integration (VII) program. Specifically, we discuss an application template for a single-user application or service similar to a commercial crash detection system, exploiting modern development tools, a Service Oriented Architecture (SOA), and illustrating a means of application distribution. Within this research, the application was developed using network services within the Google™ Android™ development platform, a data store, and a Google Earth™ service. We also outline emerging projects relating to a vehicle emission data framework and a vehicle speed retarder. It should be noted that the realization of advanced applications is not practical to deploy on scale without the momentum of the VII initiative as well as web services. Web service realization for VII applications is promising, since a uniform middleware can be achieved without extensive constraints on the underlying network and communication paradigms.

II. Vehicle Infrastructure Integration

As this paper concerns application development, limited focus is placed on outlining the lower levels of the USDOT VII initiative [1]. The research reported here extends many of the uses initially suggested under the VII initiative that were primarily oriented towards improving safety and mobility within surface transportation systems. However, it should be noted that the realization of advanced applications is not practical to deploy on scale without the momentum of the VII initiative as well as web services. Web service realization for VII applications is promising, since a uniform middleware can be achieved without extensive constraints on the underlying network and communication paradigms [1].

Early developments in vehicular telematics emphasized wireless services for individuals, such that they may make alternative route decisions in a more or less independent manner. These are essentially first generation applications, such as simple direction systems based on GPS, where the users are autonomous and assumed independent of one another. The conceptual objective of these first-generation applications is to define or identify, and then to reproduce a measured parameter (e.g. geographic location) back to the user. These early vehicular telematic applications were user-centric, whereas innovative applications that are now within reach – while still user-centric – seem to come from a 'brave new world'. It is now feasible for individuals to receive an invoice at the end of the month with several GPS/accelerometer detected infractions, such as parking, speeding, dangerous driving, non-compliant engine off etc.

The emerging – and more interesting – generation of wireless applications capitalize on the ability to extract data from fleets or multiple vehicles, such that data mining and statistical inferencing can provide better decision support of difficult-to-predict emergent behavior. The conceptual objective of these second-generation applications is to translate or convert multi-source data, and then to estimate or infer extant conditions back to the users as well as interested third parties. Examples of these applications include highway safety systems where wireless intra-vehicle relays can be used to impede or halt oncoming traffic in a freeway accident scenario (forward collision warning). These are somewhat obvious applications, built upon layers of sophisticated technology and with vehicle safety systems leading the development.

In addition to these applications, there is a class of newer applications that are cohort-centric rather than individual- or user-centric. These applications rely on inferencing from probes or floating cars, with the intent to capture the behaviours of a statistically significant portion of vehicles, such that meaningful inferences can be made and potentially generalized to the entire population of vehicles [3]. Applications include infrastructure monitoring, as well as sampling empirical data input for traffic flow control. Being

statistical in nature, in some cases, only a very small amount of floating car data is required to infer significant events such as congestion build-up or dissolution [4].

In these latter cases, data needs to be backhauled (wirelessly) and processed centrally, as it would be impractical to have all probe data processed onboard each probe vehicle. As such, this mandates the use of data collection management, operating systems, and application management systems. Once processed, relevant data can then be sent to information sinks (probes as well as third parties), informing them of road or condition hazards, for example. Because not all information mined and broadcast would be useful to all recipients, filtering applications would be required that provide the operator with useful information. For example, if a statistically significant percentage of probes detect a road hazard after central processing, only vehicles in the local (context-specific) area would need to be informed.

For applications that are collaborative in nature, it is critical that the underlying software infrastructure supports multiple mobile platforms in a ubiquitous manner. When applications are developed or modified, they need to be automatically uploaded to the probes. A similar requirement exists when operating system patches for embedded devices are required: the update has to be seamless without the requirement of individual operator intervention. Without considerable forethought into these issues, the systems being developed extempore will not scale. In previous work, we have prototyped a first generation remotely managed OS and application server [5]. The work demonstrated that these systems are complex enough to warrant high level abstractions and implementations identical to those being developed under the auspices of SOA and web services [6]. In addition, there is a need for an open architecture standard to be developed, de facto or otherwise, relieving application developers of the details of supporting layers in the protocol stack.

A. SOA within the VII initiative

Within the actual specification of the USDOT VII initiative there is little mention of web services and SOA. Its lack of inclusion in spite of overwhelming benefits is a likely a consequence of intentional parallels to early Internet development based on the TCP/IP protocol suite.

However, the second-generation VII applications discussed here would be sub-optimal if they were developed solely upon a traditional network programming environment. Applications which are run over a mobile environment most likely will have long delays and limited connectivity between peers, infrastructure, and central processing stations. In addition, these second-generation applications will most likely be developed under different incompatible platforms, making data and programmatic communications between them difficult. For example, data marshalling would be required to permit communications between UNIX and Windows based applications. Although the TCP/IP protocol suite will remain in use, applications' functionality and development can be enhanced by abstracting away from low-level dependencies and focusing rather on SOAs.

For example, second-generation (2G) VII applications can be satisfied by the SOA, with appropriate extensions. The SOA was meant to overcome the difficulties of creating distributed applications by making the data and programmatic communications between entities over a distributed network transparent to the developer. The SOA establishes a high level structure for distributed applications. There are three main entities in the SOA: Service Provider, Service User, and Service Registry. Communication between the entities is messaged based, and thus, loosely coupled. The Service Provider provides stand-alone functionality, and offers this functionality to Service Users through a message based interface. Any service that satisfies the SOA communications requirements can offer its service by registering with the Service Registry. A Service User identifies the available services by contacting the service registry, which maintains descriptions and identifications of the available services. An application developer creates a distributed application by choosing and orchestrating the appropriate services from the Service Registry. When the application runs, messages are sent to service providers, and the Service Providers respond with a desired result. The application may take a result returned by one of the services as input to another service, and so on. The SOA allows a service to send requests to other services through the same message protocol; thus, a Service Provider may play the role of a Service User.

The SOA is implemented by a trio of Web Services. The first of these Web Services can be implemented using, for example, the Simple Object Access Protocol (SOAP) [7], which implements the communication between entities in the SOA. For instance, service users send requests to service providers for some functionality by sending SOAP formatted messages to the service providers. The service providers, in turn, send SOAP formatted messages back to service users. Moreover, service providers send SOAP formatted messages to the Service Registry for registering their services. The second Web Service is the Universal Description, Discovery and Integration (UDDI) registry [8], which is used to implement the Service Registry in the SOA. The UDDI is an XML based and SOAP compatible registry, allowing entities to register themselves as services. By publishing services, the UDDI enables other entities to discover services of interest (via the SOAP protocol). The third Web Service is the Description Language (WSDL) [9], which is used by the Service Providers to communicate descriptions of their services, and by the UDDI to publish the descriptions of the services offered by the Service providers.

Fig. 1 illustrates a schematic of a basic SOA for VII applications.

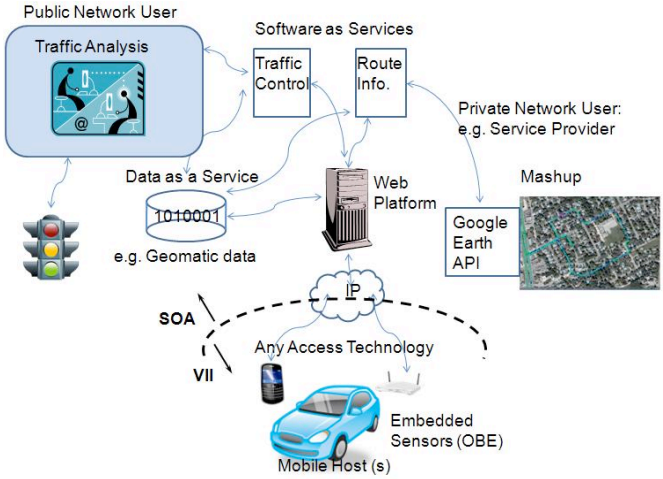

Fig.1. A VII Service Oriented Architecture.

Fig. 2. Android device running the GUI activity.

III. Current Initiatives

Our work is focussed on prototyping several vehicular telematic applications prefaced upon collaborative information gathering and/or exchange. The primary application described in this paper relates to a Vehicle to Infrastructure (V2I) prototyping environment and opportunities offered through devices such as the iPhone™ and Google's Android platform. Two emerging projects are also outlined, including an environmental emission data framework and a vehicle speed retarder. In all application development efforts, a reusable open source framework allowing for V2I applications to be developed or prototyped is of primary concern.

A. Vehicle Crash Detection Application

There are emerging opportunities in prototyping V2I applications that were not apparent even a year ago. These prototyping opportunities have been made possible through application development environments for mobiles (primarily cellular consumer electronics), and in that sense, are not specifically tailored for vehicles or ITS applications. However, they are well vetted development environments that can be reused for deploying ITS telematic applications. The two most advanced rapid prototyping platforms for third party software development and/or hardware interfacing are the Apple™ iPhone and the Android platform. There are a number of similarities, but this discussion will focus on the Android platform, as it is supported by a Linux operating system and allows for multitasking.

The application development discussed here is a variant of an OnStar™ crash detection system using the Android platform. Android is a software development platform and operating system for mobile devices. Originally developed by Google, they have since been joined by other companies such as Motorola to form the Open Handset Alliance™ [10]. Android provides a Java-based Application Programming Interface (API), making it relatively easy to get started with development and to import existing applications, although some features of the full Java environment may not be supported. The API is flexible and powerful enough to allow the application developer access to a rich set of supported device features including instrumentation (accelerometer, GPS or other location information), touch screen, network status, network sockets, video recording, and audio recording. In cases where even more functionality is required, such as direct access to Bluetooth or USB ports, the Android operating system is open source, therefore freely modifiable. It should be noted that continued development of Android is in a closed-source branch, although it is expected that new code will eventually be released.

The Android Software Development Kit (SDK) works closely with the Eclipse™ Integrated Development Environment (IDE). Installation of the Android SDK within the Eclipse IDE is in the form of a plug-in. Instructions for installation are available at the Android Developer Website [11]. Ultimately, it is useful to test designs on a real device; the effects of limitations such as device memory, as well as sensor accuracy and GPS dead-spots are not apparent in the emulator. A special Android developer handset is available, shown in Fig. 2, and it is easily configured for use with the rest of the development environment. For testing and developing vehicular telematics applications, it may be desirable to have a useful Graphical User Interface (GUI). The Android Software Development Kit includes an API for the Google Maps service, making it easy to visualize location based-information. Fig. 3 illustrates the prototypical ITS application that we implemented using the Android platform and standard Java.

For the current application, the objective is to infer when a vehicle collision has occurred, through inferencing of accelerometer and location data from the Android device (carried by the driver), and then to take appropriate action. An Emergency Dispatcher, connected to the remote CORE server using a standard Java client receives information regarding the potential collision, such as a unique name for the mobile host, accelerometer reading, location, and velocity before and after the potential collision. The dispatcher then initiates communication with the mobile host in order to ascertain whether a collision has actually occurred. Emergency services could be dispatched appropriately given the mobile host's transmitted location and status, inferred through dramatic change in velocity and corresponding deceleration.

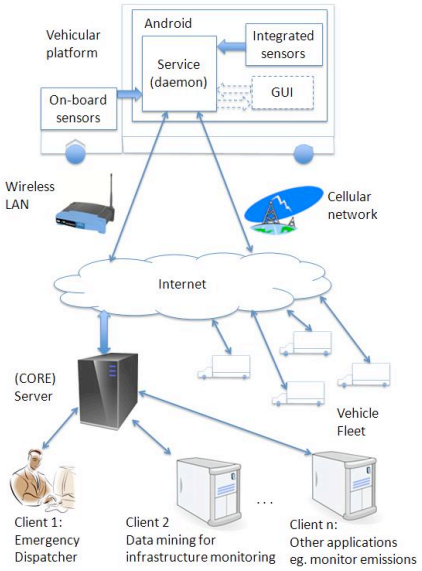

Fig.3 Diagram for prototypical Android deployment

In Fig. 3, the Android device is shown deployed on a fleet of vehicles (mobile hosts), with one shown in greater detail at the top. The application consists of a background service which monitors the sensors integrated with the Android handset, in this case the accelerometer. The service polls the accelerometer and compares the current reading with the previous reading. If the difference between the two readings exceeds a user-specified threshold, then a message is sent to a user-specified server host. Network state is available through the Android API; therefore, the Android device can implement a sophisticated policy of balancing use of the cellular network for timely delivery of data with open wireless LAN access (when available) for low cost data backhaul. In both cases, the Internet is used as a final leg to the CORE server, implemented in standard Java, where further processing takes place.

The message that the CORE server receives from the Android platform is XML formatted which should aid in later integration into a SOAP framework, if desired. The message contains information about the originating host as well as input from multiple sensors, such as the accelerometer, location and velocity inferred from the GPS. Although not implemented here, an expert fuzzy controller deployed on CORE could infer a vehicular collision by interpreting the accelerometer and velocity data. Inference data could be augmented by CORE requesting additional velocity updates from the Android device, and eventually incorporating information from OBD-II sources such as airbag deployment. (OBD-II to USB or Bluetooth to Android).

The inferencing associated with this application is likely greater than in the OnStar system. As with any new development, initial efforts need to address the potential for false positives. In part, false positives can be managed through options such as a time delay on the transmission of information to CORE server, allowing the driver to cancel the call. Alternately, the application could be developed such that the device calls itself first, and if no-one responds, the data is sent to the CORE server and ultimately the Emergency Dispatcher. Clearly, the opportunities for adding machine intelligence algorithms exist, in order to reduce reliance on management techniques to address false positives.

The flexibility and reusability of this architecture is apparent by example. By modifying the accelerometer threshold on the Android device, and replacing the Emergency Dispatcher client with a statistical inference engine, the same general architecture can be used to monitor other parameters, ranging from vehicle emissions to road infrastructure (e.g. potholes), based on data collected from fleet or probe vehicles.

Fig. 2 illustrated an early version of the prototype application developed, tested, and running on an Android developer handset. The GUI allows the user to specify the IP address and port of the CORE server, as well as allowing the user to configure the accelerometer threshold. The backend server and Google earth interface are shown in Fig.4. The backend server allows one to display locations of excess acceleration and automates a process of calling the handset once an event is triggered.

As mentioned previously, inference of an accident using course metrics that can be obtained from the Android are necessarily error prone. As such it is important that as much data that is available be considered. These are captured here as rapid deceleration, change in estimated velocity, event location, and potential call back response. Although not failsafe by any means, we present this as the type of VII applications that can be anticipated in the future.

In addition to the ability for rapid deployment, another benefit of using the existing application development environments is the access to the pre-existing application distribution network [12].

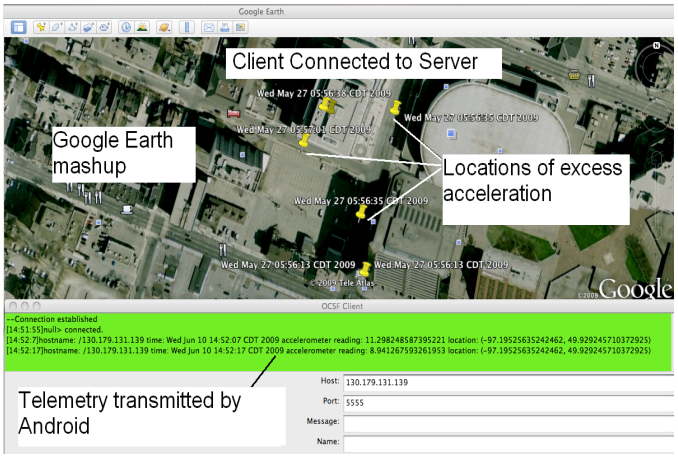

Fig.4 Server side data collection/display

B. *Vehicle Emission Data Framework*

An emerging project is the extension of the framework around OBD-II data collection over cellular or wireless hot spots in urban areas, with the objective to extract statistically relevant vehicle emission data over long periods of time and from a large number of vehicles. The project is oriented towards a real deployment using available products: OBD-II data acquisition units to collect O_2 sensor data (pre- and post-catalytic converter) [13][14], as well as the Android platform for data collection and coordinating cellular or hot spot

syncing, allowing logged data to be uploaded. The wireless uplink has been prototyped with 802.11 as well as cellular (Section III A). The prototype under development reuses others' work on an OBD-II to Bluetooth communication link from the OBD-II data port to the Symbian OS, in a GPS-enabled Nokia E71 cellphone [15]. At the time of writing, device drivers are not available for the Android's Bluetooth API but should be available by the conference date. In the future, this framework could be extended to mine additional vehicle-specific data from chassis or traction controllers, thereby providing roadway infrastructure and road condition information as well.

Beyond an interest in developing a framework with specific technologies and protocols as mentioned above, the work reflects a commitment to an open standard in which various applications can be developed, agnostic to the implementation details or underlying wireless technology. This mirrors how modern web applications are developed, where the application developer is not concerned with the details of the physical, network, transport, session, or presentation layers but rather concentrates on building the application.

An equal emphasis in the project is the development of an intelligent data collection framework, from which meaningful data mining and statistical inferencing is possible. The prototype provides a model for data collection of prolonged time-series data (spatial, environmental as well as temporal) of a large number of vehicles, upon which technology improvements and transportation policy can be assessed. Examples include inferencing the gross effect of an "engine-off at extended red lights" policy, and could be used as a means of enforcement where engine-off legislation already exists. A full-scale system would also provide meaningful seasonal statistics regarding emissions correlated to vehicle types, with similar follow-on applications to policy development. For example, in cold-climate cities where winters require a car having an engine block heater, one can envision a policy recommendation of preheating the catalytic converter, if supported evidentially.

The basic architecture for this application is shown on Fig. 5. The important aspect of the application is the reuse of the framework, extending the Android to capture OBD-II data, and to provide a means of logging large data sets for subsequent analysis.

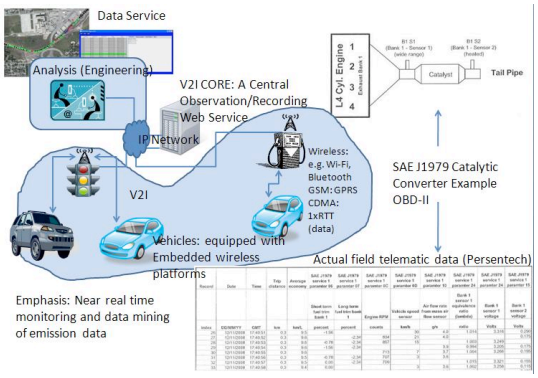

Fig.5 Emission monitoring architecture

C. *Vehicle Speed Retarder*

A second emerging project is the development of a vehicle speed retarder that is activated in designated zones (for example, school zones). This project falls within a growing group of applications related to vehicle and driver safety initiatives, including vehicle immobilizers that activate upon a given condition, vehicle immobilizers that can be activated remotely (for example, by law enforcement), and vehicle tracking technologies in which various driving parameters (speed, route, etc.) are logged and can be read from or sent to designated devices (for example, a parent's cell phone, a company's fleet manager).

In this developmental V2I application, the objective is to activate an engine retarder when the vehicle enters a designated reduced speed zone (e.g. school zone), ensuring that speed restrictions are observed. The reduced speed zone is detected either by the vehicle's on-board GPS and mapping software or by a signal transmitted by roadside hardware installed at the designated zones, as per a more standard V2I application. Examples of reduced speed enforcement range from school zones (which are of relatively long term duration) to construction zones (which tend to be much more transient and may lend themselves more easily to roadside beacons).

In the case of an Android platform interfaced to the OBD-II as discussed in Section III B, the Android's function is extended to include control signals, as opposed to simply obtaining and logging diagnostic data. Commercial OBD-II re-programmers are already widely available for automobile enthusiasts, albeit usually for performance enhancement as opposed to velocity restrictions [16]. Presumably the mode of communication between these products and the OBD-II diagnostic port is the *$08 Control operation of on-board component/system* supported by the SAE J1979 standard. In this work, we are not advocating this particular means but rather use it as illustrative of means of control. Further, we do not advocate any particular means for velocity control. It is anticipated that the mode of velocity control could be through electronic detuning of valve or injector solenoids, although alternative means of deceleration have also been presented elsewhere [17].

IV. Conclusion

This paper highlighted applications in emerging data-intensive telematic and control applications. At the application development layer, the conceptual orientation towards implementation and software development mirrors that of similar complex wireline infrastructures. In general, the applications being developed are as low-level protocol-agnostic as advanced applications being developed for the web, capitalizing on the efficiencies of layered architectures and modern software engineering.

In increasingly integrated applications, the intelligence and value are derived from a consolidation of data from disparate and non-obvious sources, as well as thoughtful approaches to data fusion and mining. Applications developed for next generation vehicular telematics will require an ITS-SOA similar to that found in the business and electronic commerce sector. Upcoming application developers will also need a healthy degree of statistical processing in his or her toolbox, to

take advantage of and extract valid inferences from collected and processed V2I data.

ACKNOWLEDGMENTS

The authors would like to thank the Canadian Auto21 Network of Centres of Excellence. In addition the authors would like to thank Huasong Cao and Dr. Victor Leung from the University of British Columbia for details on their project of collecting geocoded OBD-II data over a cellular network.

REFERENCES

[1] USDOT Vehicle Infrastructure Integration [Online]. Available: http://www.its.dot.gov/vii/

[2] R. Sengupta, C. Manasseh, and S. Levenson, "Service oriented architectures for VII – Extending web services to the roadway", [Online]. Available: http://www.its.berkeley.edu/volvocenter/VREF/UCBSOA_VIIChristian.pdf

[3] B.S. Kerner, C. Demir, R.G. Herrtwich, S.L. Klenov, H. Rehborn, M.Aleksic, and A. Haug, "Traffic state detection with floating car data in road networks," *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems*, Sept. 2005, pp. 44-49.

[4] G. Halbritter, T. Fleischer, C. Kupsch, J. Kloas, and U. Voigt, "Nationale Innovationsstrategien für neue Techniken und Dienste zur Erreichung einer „nachhaltigen Entwicklung" im Verkehr," Forschungszentrum Karlsruhe GmbH, Karlsruhe, Germany, Rep. FZKA 7157, 2005.

[5] D. Sanders, M. Laskowski, and R.D. McLeod, "A framework for mobile wireless applications," presented at Qshine WIN-ITS, Vancouver, British Columbia, Canada, 2007.

[6] Reference Model for Service Oriented Architecture 1.0 OASIS Standard, 12 October 2006 [Online]. Available: http://docs.oasis-open.org/soa-rm/v1.0/

[7] SOAP Version 1.2 Part 1: Messaging Framework, April 27, 2007 [Online]. Available: http://www.w3.org/TR/soap12-part1/

[8] OASIS UDDI Specifications TC - Committee Specifications, July 19, 2002 [Online]. Available: http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm

[9] Web Services Description Language (WSDL) , June 26, 2007 [Online]. Available: http://www.w3.org/TR/wsdl20/

[10] Open Handset Alliance™, [Online]. Available: http://www.openhandsetalliance.com

[11] Android ™ Developer, [Online]. Available: http://developer.android.com

[12] T. Yamakami, "Business model engineering analysis on mobile client-side software platform strategies," *7th International Conference on Mobile Business,* 2008, pp. 59-64.

[13] Emission Reduction Strategy using Otto [Online]. Available: http://www.myottomate.com/index.asp

[14] HUGHES Telematics Display Provides Gasoline Consumption, CO2: Application note Information Alliance™ [Online]. Available: http://www.designingwithsensors.com/applications/all_articles.php?art_id=1693&lp_id=76&nid=4681&rid=4493753

[15] H. Cao, "CellLink, Real-time data tracking of automobiles via a cell phone", BCNET 2009, https://wiki.bc.net/atl-conf/display/BCNETPUBLIC/CellLink%2C+Real-time+data+tracking+of+automobiles+via+a+cell+phone

[16] Ford Gas Edge Evolution Programmer (2004-2008) F-150, http://www.edgejuiceproducts.com/site/1622113/product/15051

[17] J.T. Zumberge, Deceleration control for automatic automotive speed control apparatus Jon Tomas Zumberge, US Patent, Number: 6535808, 2003.

Appendix B: RFID data-fusion with potential applications to Real-Time Patient Location Systems

A Network-Enabled Platform for Reducing Hospital Emergency Department Waiting Times using an RFID Proximity Location System

D. Sanders and S. Mukhi
The Canadian Network for Public Health Intelligence (CNPHI)
1015 Arlington Street
Winnipeg, MB R3E3R2

M. Laskowski, M. Khan, B. W. Podaima, and R. D. McLeod
Internet Innovation Centre
Dept. Electrical and Computer Engineering
University of Manitoba
R3T 5V6
E-mail:mcleod@ee.umanitoba.ca

Abstract

In this paper we describe a network-enabled platform for an application oriented to reducing hospital emergency room or emergency department waiting times. The platform itself is extensible, incorporating systems engineering practices and available resources, and is suitable for data collection and analysis opportunities such as those found in a networked aggregation of emergency departments or clinical settings. The content provided is described in general terms and reflects a healthcare related emergency room scenario. We demonstrate the efficacy and rationale of the platform within a service oriented architecture based framework for an RFID real time location system. Value added information mining is also indicated by extracting queuing parameters, thereby providing some degree of qualitative monitoring of service and waiting times that would otherwise be unavailable.

1. Introduction

A healthcare environment is one application area of interest that will substantially benefit from the extensive deployment of emerging wireless technology. In this scenario, the adoption of pervasive wireless communications has the potential to improve both patient safety and quality of care while enhancing overall operational efficiencies. For instance, such benefits can be realized in an emergency department (ED) setting, including a means for reducing patient waiting times.

The Network Enabled Platform (NEP) described here is focused on the dissemination of a Real Time Location System (RTLS) for the purpose of reducing waiting times and increasing operational efficiencies in an ED setting. In this deployment (denoted ED-RTLS) patients would be equipped with Radio Frequency Identification (RFID) tags or transmitters upon entering the hospital (clinic, ED or urgent care facility); similarly, personnel and medical equipment would also be "tagged" for tracking. RFID systems generally consist of tags, which are allocated to (carried by) discrete individuals, and at least one reader which senses and records the presence of any tags in its proximity.

Hospital waiting areas and associated triage areas equipped with RFID Readers (integrated within the system) would communicate with an embedded processor or thin client capable of securely collecting and uploading location and tracking data over a wireless or wired network to a more centrally located repository for further analysis.

In this application, the RFID application is moved beyond typical inventory and tracking applications, into an application for the purpose of sensing and control. The network-enabled platform is inherently collaborative and is directed toward data mining applications, wherein the data collected can be analyzed and recommendations made. Hence, statistically relevant data can be collected in the ED, in an anonymous, sanitized and transparent manner, whereupon analytical techniques (including Queuing models) can then be applied to improve ED resource budgeting, allocation, and utilization. Such a system can serve to log and correlate patient waiting times relative to their respective care providers, consultations, treatments, and even the type of equipment used in medical procedures. By using this information, statisticians and operations managers can then develop the tools necessary to aid policy makers — leading to improved patient flow and resource management in EDs. This is made possible by the identification and circumvention of anomalies and bottlenecks identified by statistical analysis and modeling of queue lengths and waiting times, along with the associated patterns of movement and interaction of patients, care providers, and equipment.

In conjunction with data collection and analysis, the information extracted can also be useful as input to simulation oriented modeling of an ED using real data. This would more fully close the loop when considering policy and policy modifications which are to a large degree done without any real quantitative support. Our use of applying notions of queuing theory to assist in modeling and understanding ED waiting times is not unique. An early theoretical attempt can be found in [1] while another attempt (that combines queuing models and empirical data) can be found in [2].

Our long term goals are to demonstrate and validate the use of RFID technology in reducing hospital ED wait times. Our objective is to infer tag (patient) location and the movement and waiting times associated with each tag (patient), using multiple readers that are networked together and deployed non-invasively throughout an ED. This is a very early and pilot stage project, in which the first step is to deploy and validate our proposed non-invasive wait time RFID-based tracking and monitoring system.

2. Applications

2.1. Local Area Patient Tracking

The application we are working on is directed towards monitoring or estimating the location of patients as they enter an ED until treated and released. This is an extremely important area of research and concern for both the general public as well as healthcare practitioners [3]. A more comprehensive description below details the ED wait time application briefly mentioned in the introduction. The local area patient tracking system, or Proximity-based Location System (PLS), allows for the development of the basic sensor technology and represents one component of the first phase of the project.

In our application scenario, a patient is tagged when registering at Emergency and a reader at the desk records the event. As the patient enters the waiting area another reader reads this event. The event and reader location are transmitted to a Hhost (Hospital host subsystem). Data would be logged by the Hhost (timestamp, tag and reader ID) and uploaded via the nearest Wi-Fi access point to be relayed to a local CORE (Central Observation and Reporting Environment). Once stored at the CORE it can be made available for mining or made available over the Internet to other stakeholders. For example, the data would be made available to IT departments of regional health authorities for further processing and subsequently could be made available to emergency planners or policy makers anywhere.

Figure 1 illustrates an ED with RFID readers deployed strategically throughout the area, capable of reading and logging patient tags for subsequent uploading to a more central network service.

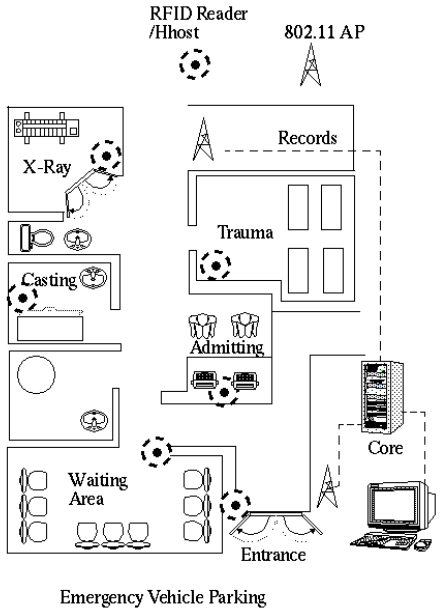

Figure 1: Schematic of tagging locations in a Hospital Emergency Facility (ED-RLTS)

In the future, it is also conceivable that a patient's tag could be updated during treatment such that a more complete record of where bottlenecks occur could be inferred. This would require the use of RFID tags capable of having data written to flash or similar nonvolatile storage. The prototype developed here uses passive tags and is best considered proof of concept. The underlying core technology however remains the same: that being, the ability to mine and infer patterns from an extensive collection of data amidst uncertainty.

2.1. Metropolitan Area ED Scenario

In a metropolitan area hospital/ED/ambulance scenario, the basic CORE functionality would scale and encompass alternative communication modalities. As CORE is designed to support wide area networking, having evolved from a geographically dispersed event management system, the task of scaling CORE is less prohibitive as it would have been if designed otherwise. As the network-enabled platform is IP centric, security issues are largely resolved using Virtual Private Networks (VPNs) and encryption standards associated with IP.

Figure 2 illustrates the basic components and agents in an extended metropolitan area framework,

incorporating several participation hospitals and emergency service vehicles. In this metropolitan area scenario, each hospital ED is still equipped as in Figure 1. Here CORE is extended to upload and log individual hospital information over a secure VPN providing basic security through encryption, authentication, and signatures. A secure web site and services can be provided for non clinical access such as via a web browser in a home or office. The data mined here would help inform people contemplating ED admittance of estimated wait times and available resources. These could include web access through traditional Internet connections at home, work, or office. In addition, cellular users could obtain similar information via cellular data services.

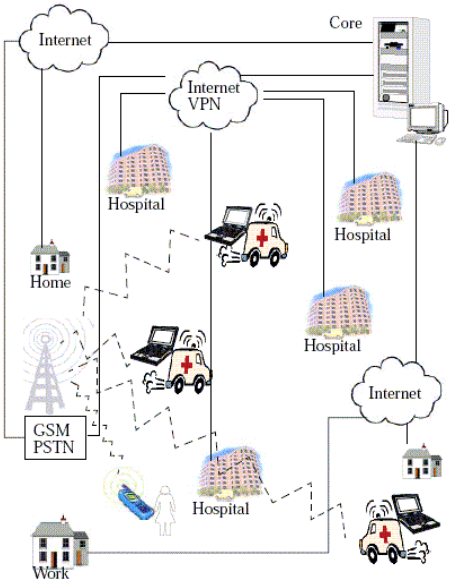

Figure 2: Wide area deployment of the framework illustrating the major stakeholders or agents.

More proactive extensions include the ability to notify and receive information from ambulances and other emergency vehicles. These likely would be over GSM or similar cellular communication infrastructures.

The metropolitan area hospital ED scenario allows for the development of Service Oriented Architecture (SOA) components that facilitate deployment nationwide as the ED - NEP. The metropolitan area aspect of the project is best considered a logical extension of phase one.

2.3. National Area ED – NEP Scenario

As the project is being designed as an SOA, the software and system requirements will be housed on an ED-NEP web site. In this manner any ED in Canada interested in deploying the system and participating in the ED-NEP would be able to download the components, and obtain thin clients and APIs such that they would be able to instantiate an ED RTLS to obtain similar statistical information and share best practices within a Virtual Organization (VO). Even in the event of not actually utilizing similar data collection services (i.e., instantiating an RFID data collection service), they would still be able to download the ED wait time models and run what-if scenarios based on statistics collected at comparable sites, thereby still participating in the VO. The scaling to a nationwide ED-NEP is best considered a later phase as it has far reaching research and development implications, requiring expanded utility and the participation of additional stakeholders.

3. System Components

This section outlines the actual technology from a web services, internet, and communication perspective. The following lists some of the primary technologies and their function.

- Remotely Managed OS: These are embedded thin clients interfaced to RFID readers and wireless technology.
- Central Observation and Reporting Environment (CORE): The SOA based CORE server is the centralized "server" that is responsible for gathering data and managing the software on each Hhost and gateways at individual hospitals. CORE is designed to scale to a nationwide system thereby allowing data collection from any participating ED in the country.
- Graphical/Geographical Information System: A GIS represents an important component if these types of applications are to be successfully deployed. The one implemented here allows a user to monitor which RFID readers are active and which applications they are running.
- The Hospital host (Hhost) Subsystem: The Hhost is an IP centric flexible architecture. This choice was selected as it represents a mature technology and allows easier integration of other IP centric technologies and SOAs.
- Security and VPNs: Security is a major concern in regard to wireless hospital infrastructure as well as for networked sanitized and personal data.
- The CORE server subsystem: This subsystem consists of several software components, including data collection, housing, analysis and presentation applications.
- Wireless: We have focused on local access through Wi-Fi 802.11 networks as they are the most readily available for experimentation and development. This however does not limit the framework as similar

applications could also be deployed over GSM and GPRS cellular networks to support data services across Hospitals or mobiles.

4. Proof of Concept

In an attempt to validate and explore the limits of the RFID system for use in quantifying ED wait times and patient movement, we needed to first demonstrate the ability to infer queue lengths and service times. Where better to demonstrate this idea than in the lines at Tim Horton's (a very popular chain of coffee shops in Canada) well worth the queues and waiting times.

Three RFID readers were set up around a Tim Horton's location at the University of Manitoba. As volunteers lined up for a coffee, one or more of the readers sensed their tag and recorded their presence: if the line is short, most likely only the reader nearest the service counter would be activated; and, if the waiting time is long, two readers would record the tag. A third reader was located at the exit of the service area, providing a means of estimating the service time as well. The data was then uploaded over an 802.11 wireless network and logged. In addition to relying on RFID tag monitoring, and for purposes of statistical analysis and building our inference engine, the actual queue length was monitored remotely over video.

The trials were run on March 17 and 18, 2008. We were able to collect enough data for valid statistical processing, analysis, and data mining in order to infer waiting times or queue lengths, thereby demonstrating the initial conjecture and utility.

Approximately 200 RFID tags were used with approximately 1 in 3 customers carrying a tag. Figure 3 is an illustration of the process. A volunteer would arrive in line, chat with friends and patiently wait for their turn in line. Readers (not quite like the ones in the illustration) recorded position, ID and time stamp data. The working assumption was that if a sufficient number of tag-carrying volunteers are in the line, one should be able to estimate the expected waiting time any given person would experience.

The video capture tool was a shareware tool called "Webcam Uploader 2004 v4.0.6" [4]. The RFID readers used were the GAO-90 (125 KHz. - low frequency), medium range RFID readers. The RFID tags were 125 KHz low frequency Clamshell RFID passive cards.

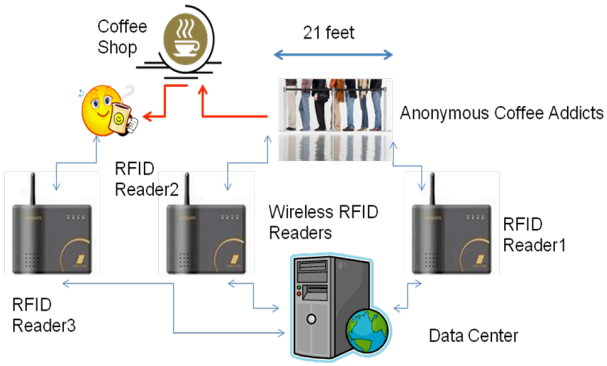

Figure 3: Anonymous queue length estimation at Tim Horton's coffee shop.

5. Inference Engine and Results

One of the first pieces of data extracted from the RFID readers was an estimate of the service times. These were compared with estimates taken from the video monitoring as well as separate estimates.

The average service time, taken from video and actual data samples, is approximately 35 seconds. This is in good agreement with the RFID data samples where the service time typically includes two customers being serviced (due to the location of the reader nearest the front of the queue). These types of heuristics and inference corroborated by alternative modality will be required for use within any clinical setting.

Figure 4 illustrates the service time extracted from the RFID readers. The actual queue length or more appropriately, the waiting times, have to be inferred from multiple readers and an estimate of service times. The inference engine heuristics used here are illustrated in the pseudo code of Figure 5.

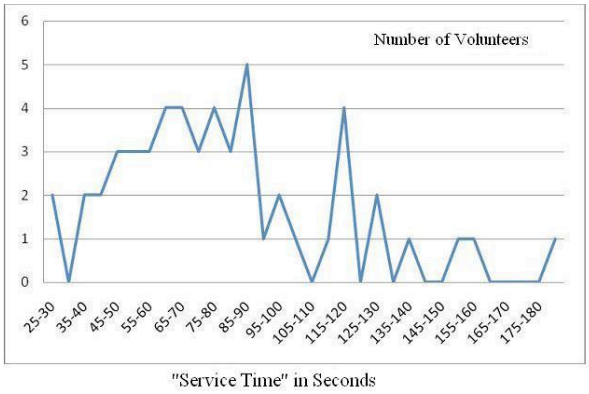

Figure 4: Service-time distribution extracted from RFID readers.

```
While{true}
  estServiceTime(ID) =
    lastReadReader2(ID) - firstReadReader3(ID)

  estWaitingTime(ID) =
    max{
    lastReadReader2(ID) - firstReadReader2(ID),
    firstReadReader1(ID) - lastReadReader2(ID)
    }
```

Figure 5: Pseudocode for estimating service and waiting time.

The calculation of waiting time can then be compared with the actual monitored queue lengths. Of course there is not a perfect correlation as there are only a limited number of readers. (More readers would have provided finer proximity estimation quantization and hence a correspondingly better approximation to actual waiting times.) The "estimated" waiting time represents a weak lower bound. Figures 6 and 7 represent the actual queue length and the queue length distributions, respectfully, extracted from video surveillance for March 17.

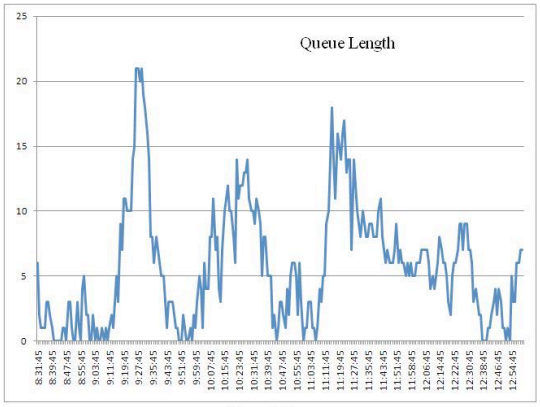

Figure 6: Actual queue lengths March 17.

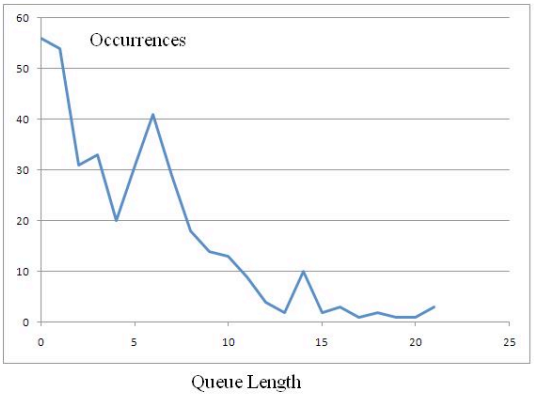

Figure 7: Actual queue length distribution March 17.

If the waiting and service time distributions can be inferred from the RFID data they can be made available for use within simulators that could be used by operations researchers and policy makers to improve emergency department efficiencies. A difficulty without using technologies such as those presented here is that models based on known distributions are not likely those that best describe the underlying stochastic processes, supporting the case for utilization of empirical data if available.

Given the actual queue lengths and estimated service times of approximately 35 seconds, the waiting times can be estimated. Although not obvious from Figure 6 the arrival process is non stationary.

On March 18 a similar study was undertaken. In this case however no new tags were provided to customers. The overall number of customers was comparable, but there were approximately 30% less customers with tags from March 17 (the day before). Although "coarse," this allowed for some degree of appreciation of the sampling that would be required to infer meaningful information collected through statistical sampling.

Figure 8 illustrates the estimated lower bound for queue lengths extracted from the RFID data collected compared to the actual queue length.

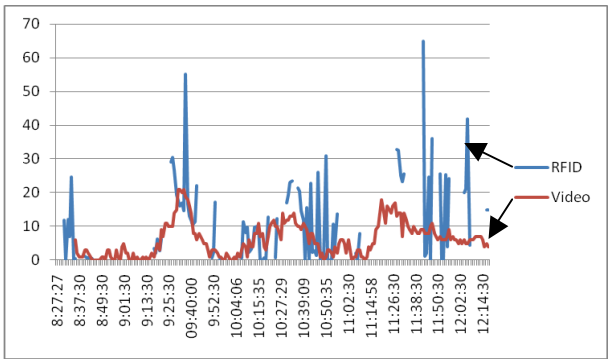

Figure 8: RFID inferred waiting times in sec/10 (non contiguous data).

5.1. Analysis

Our initial conjecture was that by using an RFID proximity-based location system coupled with statistical inference, we would be able to extract useful parameters such as waiting times. Although not obvious from Figure 8, the estimated RFID waiting times are well correlated to the actual queue lengths monitored, with a correlation coefficient of 0.533. This correlation coefficient is maximized at a time shift of approximately 5 minutes delay (of the video queue monitoring). The time shift accounts for the fact that the video queue

length data is extracted in real time; whereas the time stamp associated with the RFID data is for when the person is at the head of the queue — which coincidentally provides another estimate of the waiting time. The average waiting time extracted from RFID data is approximately 2 minutes whereas the estimated waiting time from the queue length monitoring was approximately 4 minutes. This discrepancy is partially accounted for by the location of the RFID readers. Once the line was longer than 8-9 customers they were out of range of the readers as illustrated in Figure 9.

Figure 9: Video snapshot of queue and RFID reader locations and read proximity.

In addition, the reader at the head of the line was actually located nearest the 3rd or 4th customer. As such, the waiting times inferred by the RFID readers should be less than the actual waiting time. (This also accounts for differences in the actual service times.) Thus, accounting for these differences, and the service times inferred as opposed to those measured, the RFID waiting time would be approximately 3 minutes (augmented by 30 seconds extracted from the RFID service time overestimate, as well as the 30 seconds of waiting time unaccounted for by lines longer than 10 customers). The findings illustrate the necessary prerequisite for successfully inferring queuing model parameters using proximity-based RFID location systems. This implies that data is to be mined with a sufficient statistical ensemble size (and sampling), along with the strategic spatial positioning of RFID readers, to attain additional information beyond inventory types of applications.

6. Summary

Most efforts to date concerning RFID are related to inventory and tracking. Emerging RFID technologies are providing the opportunity to deploy applications that incorporate sensing and control. Our work here extends RFID technology from the data mining perspective – that is, statistically mining tracking data to infer queuing parameters important for processes that are amenable to queuing systems. The pilot stage project provides preliminary validation of an RFID-based tracking and monitoring system in a queuing environment. The experiment carried out and presented here serves as a proof of concept in validating our conjectures of the utility of using RFID technology to estimate queuing and waiting time parameters.

An ED network-enabled platform was presented here aimed at reducing ED wait times through the use of extensive — albeit non invasive RFID sensors. At the lowest hierarchical level there is an extensive distributed wireless hospital RTLS environment oriented to patient tracking via RFID in EDs. At the next level there is a metropolitan architecture based on best SOA practices for data collection, analysis, and presentation. The system we are developing is open source and relies on open source components. It is therefore readily extendable and can be ported or tailored to a variety of IT applications beyond the one identified here.

7. References

[1] D. J. Worthington, “Queueing Models for Hospital Waiting Lists”, The Journal of the Operational Research Society, Vol. 38, No. 5, May 1987, pp. 413-422.

[2] H. Piene, P.A. Nyen, H.K. Hauge, “Waiting lists and hospital capacity - analyses based on theoretical queue models and empirical data”, International Society of Technology Assessment in Health Care (Annual Meeting), 1997, 13: 47.

[3] S Trzeciak and E P Rivers, “Emergency department overcrowding in the United States: an emerging threat to patient safety and public health”, *Emerg. Med. J.*, 20, 2003, pp. 402-405.

[4] “Webcam Uploader 2004 v4.0.6” – available from http//www.download.com.

8. Acknowledgements

We would like to acknowledge the support of the Canadian Network for Public Health Intelligence (CNPHI), IDERS, Virtuistix Inc., the Tim Horton’s service staff in the Engineering building at the University of Manitoba, as well as the approximately 200 coffee “addicts” who voluntarily carried RFID tags during the proof of concept trial. We would also like to

thank Professor Attahiru S. Alfa for his support, both financial and intellectual.

Appendix C: ABM as a Tool for Communication between Healthcare practitioners and Modelers

Educational Opportunities in Agent-Based Modeling

M.R. Friesen, R.D. McLeod, M. Laskowski, and M. Borkowski

***Abstract*— This paper outlines opportunities for student work in agent-based modeling. It overviews applications ranging from community-level epidemic modeling, epizootic modeling within a livestock production operation, modeling patient waiting time in a hospital emergency department, and modeling the spread of nosocomial infection within a hospital. The modeling applications are presented to demonstrate the educational opportunities inherent in agent-based modeling, which can address the educational and professional context calling graduate engineers to have a wider range of professional skills, including communication, teamwork, self-management, creativity, and awareness. Agent-based modeling applications lend themselves naturally to project-based work, to interdisciplinary exposure, to professional skill development, and an association with engineering priorities at a societal level. Agent-based modeling is also supported by concepts in teaching and learning theory.**

***Index Terms*—agent based modeling, engineering education, interdisciplinary education.**

I. INTRODUCTION AND BACKGROUND

A pervasive theme in engineering education is the need for the curriculum to be proactively innovative and adaptive, in response to a rapidly-changing world. This theme translates into calls for an engineering curriculum that increasingly facilitates a wide range of professional skills in graduate engineers, including communication, teamwork, self-management, creativity, adaptability, social awareness, and ethical leadership [1][2][3]. Among its curricular outcomes, ABET lists the need for graduate engineers to understand contemporary issues and to understand the impact of engineering solutions in a global, economic, environmental, and societal context. The theme is also manifest in the acknowledgement that the boundaries between engineering disciplines, once well-defined, are becoming increasingly blurred. In practice settings, engineers are required to work in interdisciplinary contexts with professionals of other disciplines [3][4][5]. This is directly evident in the increasing emergence of undergraduate programs that are explicitly integrative, such as biomedical engineering, as well as the struggle to balance core courses in a curriculum with persistent calls to accommodate more social sciences, management, and other courses in the undergraduate engineering program.

In addition to body of knowledge requirements set forth by accreditation bodies, various engineering disciplines have advanced efforts to articulate a body of knowledge for graduate engineers in their disciplines [6][7]. The IEEE Computer Society has also articulated such a statement [2], which focuses heavily on discipline-specific topics, although inclusive of the social contexts, professional, and ethical responsibilities sought in graduate engineers. Within the delivery of the undergraduate engineering curriculum, many of these objectives and emerging realities are addressed through team-based, project-oriented work, which are intended to foster professional skills and an inter-disciplinary orientation (for example, [8][9][10].

However, within the broader scope, the application areas of professional engineers are expected to shift. The National Academy of Engineering [3] has outlined various scenarios that provide both opportunity and challenge to engineers in the future. These scenarios include the new scientific revolution, in which new technologies are developed and optimized to the benefit of society; the biotechnology revolution; and, the natural world and engineers' contributions to help predict risk and design adaptive systems in the face of natural disasters. In all of these scenarios, engineers interact with societal attitudes and politics as they negotiate technical possibilities to address challenges of infrastructure, environmental issues, as well as housing, water, and healthcare for a changing and growing population [11].

In Canada, the first annual National Engineering Summit took place in 2009 [12], organized by the Canadian Engineering Leadership Forum bringing together the general public and six national engineering associations representing engineering industry, education, and regulatory bodies. The summit focussed on health, environment, safety and security, competitiveness in the global economy, and quality of life. The purpose of the summit was to challenge the engineering profession to develop a cohesive vision of the profession's involvement in and contribution to these areas going forward. In relation to the undergraduate engineering curriculum, enhancing inter-disciplinary opportunities and curricula emerged as a strong theme at the summit.

M.R. Friesen is with the Design Group, University of Manitoba, Winnipeg, MB, Canada R3T 5V6 (e-mail: marcia_friesen@Umanitoba.ca)

R.D. McLeod is with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada R3T 5V6 (e-mail: mcleod@ee.umanitoba.ca)

M. Laskowski is with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada R3T 5V6 (email: mlaskows@ee.umanitoba.ca)

M. Borkowski is with Microsoft, Seattle, WA, USA (email: macbork@microsoft.com)

Within computer engineering, the application areas of technical knowledge have consistently shifted over the past decades. In the 1980s, integrated circuits were a key application in computer engineering. In the 1990s, the focus shifted to telecommunications and the internet, and presently, new opportunities for computer engineering applications are emerging. Whereas descriptive language may change – for example, from charges and gates (integrated circuits) to routers and packets (telecommunications), some foundational constructs remain the same. In all of these applications, computer engineers are concerned with structure and behaviour, and with modeling and simulation of the same. These foundational constructs are consistent and can be redeployed into new application areas with new descriptive languages.

Within this educational and professional context, this paper outlines efforts in agent-based modeling that continue to build on the constructs of structure and behaviour, in this case, of human and animal agents. The projects outlined in this paper are demonstrative of the emerging application areas of engineering; discrete project entities appropriate for capstone design, thesis research, and graduate work; the increasingly interdisciplinary nature of engineering education and practice; and, professional skill development in students. As such, these projects represent opportunities to creatively and proactively adapt computer engineering education, and to develop knowledge and skills in graduate engineers that reflect emerging industries and labour market needs.

II. AGENT BASED MODELING

There are a large number of traditional mathematically-based approaches to system modeling [13][14][15]. The majority of these mathematical approaches are quite dissimilar to agent based models / modeling (ABM). In ABM, a system is modeled from the ground up: describing a system from the perspective of its constituent parts (microscopic view), in order to build an aggregate (macroscopic) picture of the whole. Systems are modeled as a collection of agents, each of which are autonomous decision-making entities able to assess their situation, make decisions, and compete with one another on the basis of a set of rules. In that way, ABM can be described as system modeling based on individual agents and their behaviours and interactions, yet an ABM's conceptual depth is derived from its ability to model emergent behaviour that may be counterintuitive or, at minimum, its ability to discern a complex behavioral whole that is greater than the sum of its parts [16].

The benefits of ABM lie in the technique's ability to capture emergent phenomena, which also leads to further benefits: ABM provides a natural description of a system that can be calibrated and validated by representative expert agents, and ABM is flexible enough to tune the complexity of agent behaviour, rationality, learning, and rules of interactions [16]. There are a number of characteristics of agents and systems for which ABM is a particularly powerful modeling technique, and which simultaneously make population-level equation-based approaches difficult or intractable [17][18]. These system features include individual agent behaviour that is complex, non-linear, potentially stochastic, and may exhibit memory or path-dependence, agent interactions that are heterogeneous, instances in which averages are not meaningful, and systems that are best described by activities and interactions rather than by processes.

An ABM allows for near-universal application, and confers an advantage by its naturalistic (vs. abstracted) means of knowledge translation and a naturalistic means of implementing agents. That is, the specification of the problem from the practitioners' perspectives is identical to the specification and requirements capture in the engineering modeling process, without intermediate levels of abstraction. It is the intermediate levels of translation and abstraction that add error to the process, and thus a technique that minimizes the hierarchy of problem translation specification is favorable.

Considerable work has been done with ABMs in the area of community-level epidemic modeling in human populations, as this is an important unsolved problem which garners a significant amount of public health and public policy attention [19][20]. Furthermore, ABMs have also been applied to logistics, economics, business, and tactical decision making and support systems. In these and other applications, ABMs allow one to identify losses and test mitigation procedures, and model risk in general [16].

The opportunities outlined below, situated in electrical and computer engineering and computer science, are primarily related to public health applications and veterinary health applications. Thus, they are by their nature interdisciplinary, which is further highlighted in further sections. As well, the opportunities capture unique combinations of agents characteristics, institutions, and objectives to simulate critical incidents as well as the dynamics of routine interactions. In the examples, the ABM framework is coupled with additional context-specific frameworks that enhance the technique's potential, including a Discrete Space Scheduled Walker paradigm, percolation theory, and machine learning.

In each case, the objective of the ABM is to generate both descriptive and prescriptive outcomes. Descriptive outcomes are to gain an understanding of the phenomenon of interest, such as the dynamics of disease spread within a population. Prescriptive outcomes are to gain an understanding of the relative impacts and sensitivities of the input parameters – that is, to model comparative 'what-if' scenarios that can provide empirical support to policy and practice decisions related to the phenomenon of interest.

The following section outlines our current work in ABM development, as a means to provide the technical context for a discussion of the educational opportunities inherent in the work. The ABM development is applied to unique configurations of agents (humans, animals, and inanimate objects) and institutions (single and multiple, as well as community-level modeling), with a view to simulate critical incidents as well as routine interactions. Ongoing objectives are to extend and refine the developmental ABMs for each specific application, in order to enhance their capacity to

provide insights into system dynamics, model risk, and assess comparative mitigative strategies.

The first of the ABMs is applied to modeling the spread of a disease such as epidemic or pandemic influenza in a general population, and the role an ABM can play in disaster response in a general population. The second example illustrates the application of an ABM to epizootic modeling, whereby an intensive livestock production operation is modeled. The third example is an ABM modeling patient access and patient waiting times within a hospital emergency department, with natural extensions to modeling the spread of hospital-acquired infections (nosocomial infections) within healthcare facilities.

III. ABM APPLICATIONS

A. Epidemic Modeling

Epidemic modeling is an important unsolved problem with significant interest from public health and public policy agencies, and thus warrants research not only from these fields, but from empirical and engineering perspectives as well. As such, one of the cornerstones of ABM is its multidisciplinary nature, where – at minimum – agents' behaviours and interactions are sociological inputs derived from appropriate content area experts.

Work to date has focussed on developing and validating a proof-of-concept to model the spread of disease in a medium-sized North American city, with additional detail available in [21]. Many similar approaches are used for modeling the potential spread of disease between cities, the work here is differentiated in that it is a intra-city model. Further efforts are focussed on extending the work with the integration of real demographic and behavioral information data, specific to a particular city.

The ABM approach applied to epidemic modeling is based on the paradigm of a 'discrete space scheduled walker' (DSSW). The DSSW ABM approach encompasses "where, who, when, and what", built upon a conceptual framework of common notions and relatively simple statistical reasoning, such as the law of large numbers and statistical mechanics, as well as a correct-by-construction bias. **Where:** Underlying topological (network/graph) data is extracted from map utilities, land-use databases, and search engine utilities, building a network of objects denoted "institutions". Institutions are existing geographic locations such as homes, businesses, leisure sites (malls, restaurants, etc.), schools, universities, hospitals, airports, cars, public transport, etc. in which agents would spend time. **Who:** Agents are the people (children, teenagers, and adults) that make up a community, and between whom a disease would be spread. Agents' behaviours (patterns of movement, etc.) and interaction are based on rationally-inferred behaviours extracted from demographic information and applied to a large percentage of a population. Within the DSSW ABM, each individual person is modeled as a discrete agent. **When:** A central premise of the DSSW ABM approach is that agents are primarily creatures of habit, largely operating on routine schedules with slight perturbations. As such, the approach shares conceptual similarities to geographic profiling used in law enforcement, which is in turn more fundamentally built upon routine activity theory and rational choice theory [22]. **What:** The "what" of interest in the DSSW-ABM approach is the spread of viral or bacterial disease such as influenza. Influenza is "who-agnostic" – that is, it is contracted by contact associated with routine daily activities, and every agent is more or less equally susceptible.

Many previous ABMs attempt to build random graphs or small world networks which would then arguably represent a city, a collection of cities, or a region. To some degree the interconnection models of cities is being replaced by the actual transportation networks, e.g. air travel and overland movement of goods. In contrast, there has been concomitantly less isomorphic mapping for a network of people *within* a city. Our work here is one of the first to exploit correct-by-construction inter-city topologies extracted from Google™ Maps or Earth.

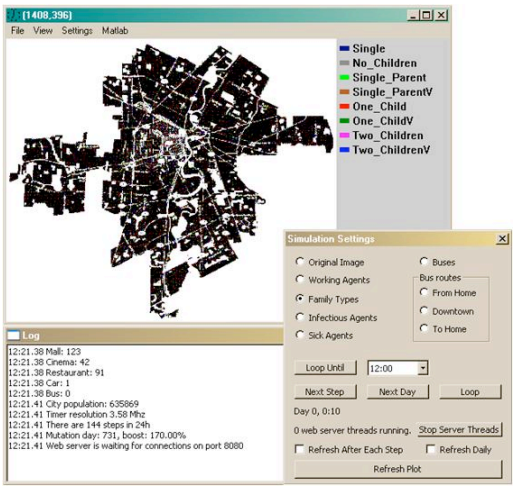

Fig. 1: The Graphical User Interface of the DSSW Simulator

At present, the DSSW ABM proof-of-concept has an interface and associated scripts that allow for a variety of comparative what-if scenarios to be demonstrated and for data to be collected and displayed via a web browser. An example of part of the user interface is illustrated in Fig. 1. In this scenario, Winnipeg, Canada as an instantiation of a medium-sized North American city is modeled, using approximately 650,000 discrete agents. The topography was extracted from Google Earth, whereas the population demographics were simulated, varying the distribution of family units and types across the city. In this DSSW ABM, the effect of various public health policies (vaccination strategies, etc.) were simulated, similar to those currently being proposed by the CDC [23]. An ABM more easily facilitates this type of simulation than other techniques. Furthermore, the DSSW ABM proof-of-concept has number of useful presentation capabilities, including a web server that houses the results of the simulation as graphs and raw data (MS Excel format) suitable for follow-on processing. This latter aspect – while obvious – is crucial to an overall ABM framework developed within an interdisciplinary team.

A natural extension is to community-level epidemic modeling is to model community-level preparedness planning, mitigation during, and recovery from a natural disaster. In this

case, the "when" and "what" input parameters are modified and extended to the new instance. Just like the underlying DSSW ABM for epidemic model is "who-agnostic", this application is also "disaster-agnostic" in that the framework is flexible enough to handle the characteristic parameters of a number of specific natural disasters. Within the same, the model can handle multiple schema ("what if" scenarios) to again determine the relative impacts and sensitivities associated with individual inputs.

An immediate and obvious priority during a natural disaster is the maintenance of critical infrastructure (both physical and service infrastructure) and the provisioning of goods and services during a disaster. These include, but are not limited to food, water, pharmaceuticals, fuel, energy utilities, public services such as healthcare, and transportation logistics. An ABM with sufficient fidelity could be used to model the rationing or planned degradation of critical utilities and services, if measures called for such.

B. Epizootic Modeling

While much ABM work to date has focussed on agents as people, and on modeling of disease spread and its effects, similar concerns are present in epizootic modeling. Our research to date has focussed on the intensive livestock production operation (ILPO, at times called the 'factory farm') as the instance context. The ILPO offers a well-defined and well-controlled environment for modeling, and is representative of the conditions in which livestock are produced. The genetic homogeneity present in an ILPO also presents a risk factor in terms of rapid disease spread. ILPOs are an integral part of the food production system in North America, with very significant direct and indirect economic impacts, and thus preventing and mitigating disease spread within ILPOs are key concerns of livestock producers.

The work has considered the role of ABMs in modeling the spread of disease within an ILPO, illustrating the role of agent mobility in propagating disease within a poultry ILPO when modeled as a percolation problem [24], and further efforts are aimed at extending the model to a swine ILPO. The objectives are to gain insights into the impact and sensitivities of changes in livestock housing design and animal management practices within an ILPO. Examples of the former include room and facility configurations, equipment installations, and biosecurity measures, and examples of the latter include animal flow, population segmenting, and quarantine measures representative of ILPOs. These are modeled to demonstrate their relative impacts on epizootic spread and to derive mitigative strategies.

Within the study of disease spread within a poultry ILPO, the ABM approach was combined with a model of behavior extracted from percolation theory, augmented with notions of agent mobility. These types of ABMS are the most simplistic in that there is little decision-making capabilities afforded the agents; however, their interactions are of primary importance. The objective of the work was to develop and validate a behavior-based disease modeling framework, and in doing so, to examine the relative impacts of longevity and mobility on epizootic spread and animal mortality. The ILPO was modeled as a square lattice with a site percolation threshold of approximately 0.42 for a neighborhood with both adjacent as well as diagonal neighbors (Moore neighborhood). By introducing agent mobility, a key characteristic of the modified percolation model was that nodes are not only able to affect their adjacent nodes, but nodes also had the ability to move on the lattice. This introduced a variant more closely associated with cellular automata than traditional percolation theory. While the model was coarse with identified shortcomings, the attempt was to develop a model that captured critical phase transitional phenomena, while remaining relevant to the instance context and the ABM framework.

The study focused on a coarse model of a poultry broiler barn, representative of the simplest possible ILPO topography (a defined, single-space environment). The study modeled agent mortality with the following independent variables: population density, corrected to population on an invariant square grid; mobility of the agent (chicken), correlated to a probability of agent movement in a combined x and y direction, to an adjacent node on the grid; and, longevity of the contagious (infectious) period, equated with the duration of disease manifestation in the animal, an also equated with the duration of animal mobility and longevity. A graphical user interface allows the user to vary the independent variables, as well as to define the size and location of the initial infection area on the grid.

Systematic analysis investigated the impacts of longevity and mobility under different populations. Fig. 2 displays the general shape of the mortality results for ranges of agent mobility and agent longevity values (in simulation cycles), for a given population value. While the transition from low mortality to high mortality is quite steep (i.e. sensitive to small changes at specific threshold values of the independent variables), it is also worth noting that for the majority of input conditions, the disease remained enzootic rather than epizootic. The Beta coefficients for longevity and mobility against population indicated that at low population density, the longevity of the infectious period has the greater impact on mortality, while at high population density, the impact of agent mobility dominates.

An ABM for disease spread in a swine ILPO would introduce a considerably more complex topography and a greater range of animal management and animal care practices. While a swine barn is still built around predictable and symmetrical compartmentalization, the swine ILPO represents a greater variety and a more complex nature of building layouts (rooms and penning), animal equipment installations, bio-security measures, and animal flow and management practices. In either case, modeling could also be extended to interactions between multiple ILPO sites.

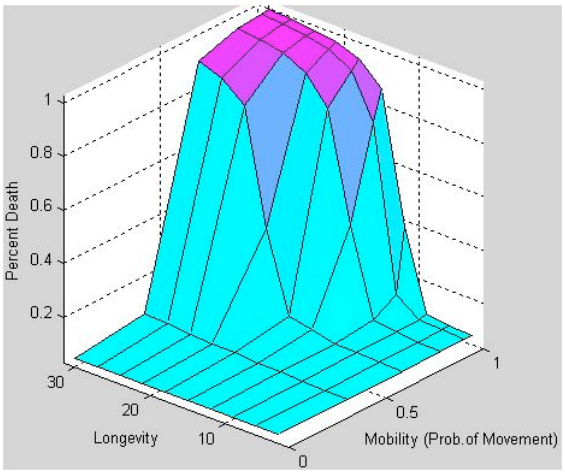

Fig. 2: Representative Simulation Output

C. Modeling Patient Access and Wait Times within a Hospital Emergency Department

Whereas the previous examples have focussed on modeling a critical incident within a city (a network of institutions) with human agents, and modeling a critical incident within a single institution with animal agents, the following model is an extension that focuses on a single institution with human agents. By focusing on a single institution, the work focuses on all input parameters to a high degree of accuracy, based on contextually-validated assumptions, to a high degree of parameter sensitivity, and with a high proportion of real data inputs. This research moves toward improved understanding and management of patient access and wait times within an emergency department (ED) or urgent care clinic – in particular, to evaluate workflow within one ED (patient wait times, service times, and staffing policies) and assessing patient diversion policies between multiple EDs. Further details are available in [25][26].

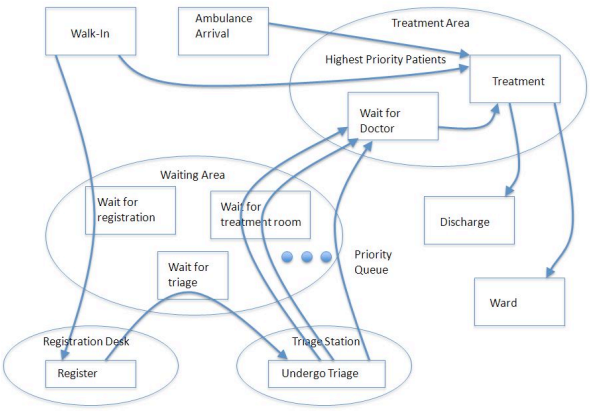

Fig. 3: Model of Emergency Department Patient Service

The ED processes of interest to the patient access ABM are shown in Fig. 3. The work relies on a general model of an ED floor plan. The scenario investigated, while simple, illustrates the effects of policy decisions, such as changing staffing levels, using multiple performance metrics. In simulations of varying staffing configurations, the basic ED workflow is simulated, with Triage Classes, Service Times, and Patient Arrival rates based on Patvivatsiri [27]. Three different staffing scenarios of two, three, and four doctors on duty are compared. Ten independent trials were run; average treatment queue length is shown in Fig. 4. Alternatively, doctor utilization or individual patient waiting times can also be instrumented. The staffing simulation is shown as representative of the modeling capacity, and intuitively the results appear to validate the model.

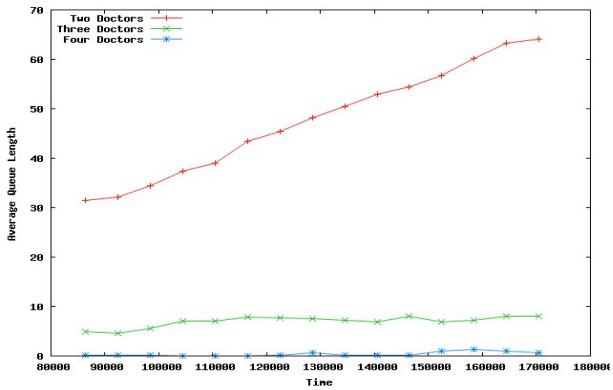

Fig. 4: Average Queue Lengths for Varying Number of ED Doctors on Duty

In addition, ABMs can incorporate evolutionary algorithms that allow realistic agent learning, and extensions of this work include the addition of a Machine Learning (ML) module with the ABM framework to provide for automatic policy generation. The ML module generates policies, uses the ABM to evaluate them, and then uses the best individual policies as the basis for the next generation of policies. This process is iterated until pre-defined criteria are met. Genetic Programming (GP) [28] is a ML paradigm for the automatic induction of computer programs through an evolutionary process. The GP paradigm is well established and includes successful research applications in the areas of data mining, image classification, automatic circuit evolution, and robot control. Evolutionary Algorithms (EA), a group of algorithms to which GP belongs, can improve upon human generated policies, and sometimes in unexpected ways [29].

The opportunity exists to use the ABM to investigate the viability of using a GP-based ML system to forecast ED waiting times. As data from the data collection frameworks become available [30], the ML system could be trained and validated on real data. In this instance, the ABM becomes a data generator and an input into the overall ML paradigm. Fig. 5 illustrates how the ABM provides feedback to the GP-based ML system.

To accommodate the automated policy generation task, refinements to the model parameters are required. The refinements encompass the agents – for example, the ability of a patient to change its internal state probabilistically to get less or more seriously ill or leave the ED, and then evolve triage policies to optimize patient flow for these more complex agents.

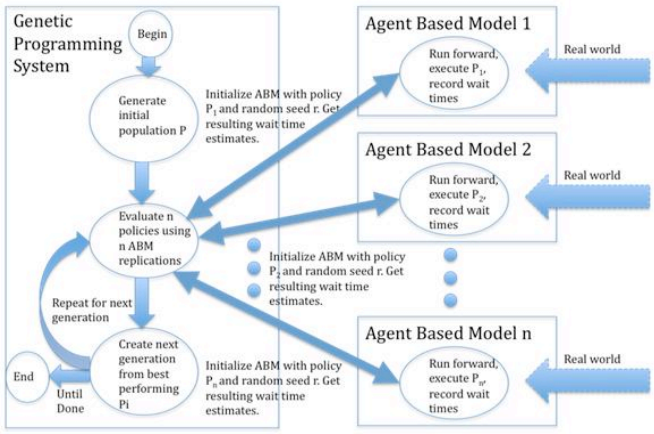

Fig. 5: GP and ABM Integration

A second refinement may be to generate an architecture-agnostic patient diversion policy, where only the policy is evolved, and the means of implementing the policy is assumed to be in place and is treated as abstract. Third, an agent class responsible for executing patient diversion policies generated by the GP-system can be added to the ABM.

Finally, a further opportunity is to use ABMs to model alternative forms of ED care. There are emerging efforts to reconceptualize the ED around a main function of addressing emergencies (vs. current models of maximizing patient flow). This has follow-on effects in multiple directions, including staff configurations (teams vs. individuals) and the physical layout of the facility. Here ABMs offer a useful tool in investigating the impacts around such paradigmatic shifts.

D. Hospital ABM Extensions to Nosocomial Modeling

In this application, the institution is a hospital or medical clinic, where the ABM is applied to model the spread of hospital-acquired (nosocomial) infection throughout the institution. The topography and agent movement inputs are held in common with the previous example relative to patient flow in an ED. Agents would be modeled for their patterns of movement and interaction, and the topography – the physical layout of the institution – would be modeled with sensitivity to the varying infection probabilities associated with different areas of the hospital. Areas with high disease spread potential (e.g. emergency department waiting areas, intensive care units) would be uniquely characterized within the overall model. Fig. 6 represents the basic extensions of the patient access ABM to allow for the modeling of nosocomial infections, and Fig. 7 illustrates the visualization of the ABM for nosocomial infection spread. Further details are available in [31].

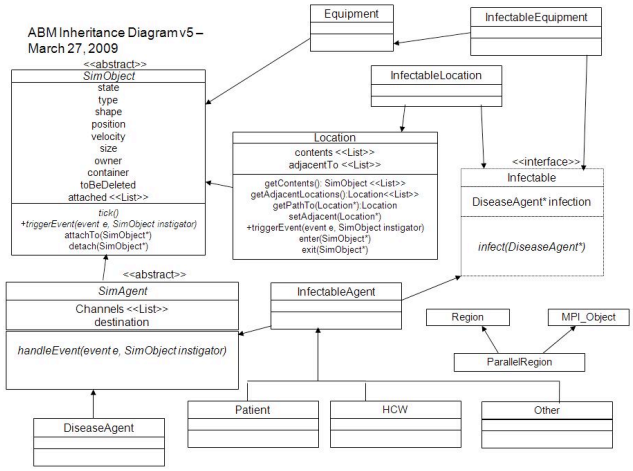

Fig. 6. Inheritance Diagram of the Patient Access Augmented to Model Nosocomial Infections

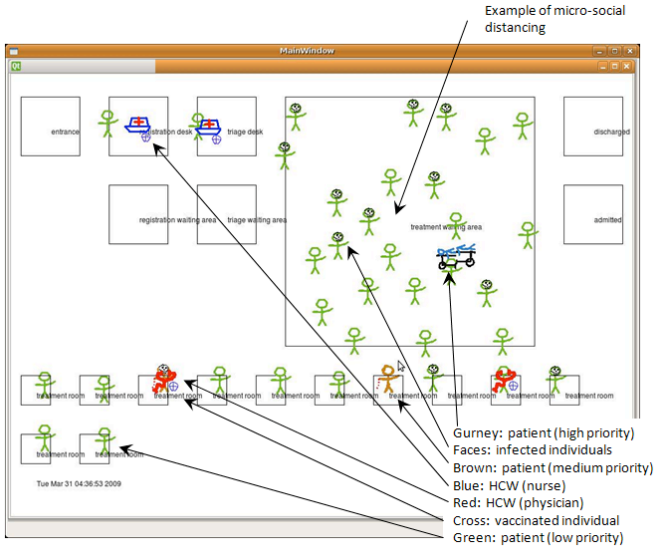

Fig. 7: Visualization of the Nosocomial ABM

Many of the ABMs oriented to nosocomial infections are known as "individual based models", in which agents are limited by definition to individuals (persons), (for example, [32]). By contrast, our notion of an ABM is notable in that it expands the definition of agent beyond an individual person, to include inanimate objects that can act as vectors of transmission for nosocomial infections, such as medical equipment and supplies, and defined locations. This concept is supported by a considerable body of evidence that non-person agents play a significant role as infection transmission vectors [33][34], including the CDC's overview of SARS related information [35].

IV. EDUCATIONAL OPPORTUNITIES

ABM development presents a unique combination of educational opportunities for both undergraduate and graduate level work in electrical and computer engineering and in computer science. First, ABM is a natural application to the emerging application areas in engineering. Whether focussed on health, environment, public safety, food security, or other

social concerns, current priorities are systems-level applications and thus inherently suited to ABM.

Second, ABM development naturally lends itself to project-based work, in which individual projects address distinct component of the ABM. Because of their fundamental nature of being built from the ground up, components of the ABM development toward any application can be discretized. This applies to the ABM's topography, its agents (types of agents, characteristics of agents), as well as the behaviour of and interactions between agents. These parameters can be moved from coarse-grained, simple approximations to increasingly refined states of development. As models are extended and validated, the additional classes of agents and institutions are added and the nature and range of agent behaviours and interactions are augmented – for example, replacing binary variables (such as, infections – non-infectious) with statistical distributions (such as, a probability of infectiousness). Technical complexity is also achieved by coupling ABM with other technical frameworks, such as percolation theory and machine learning. This range of complexity allows ABMs to find applications ranging from short-term undergraduate projects to long-term graduate work.

Third, and most significantly, ABM development is inherently interdisciplinary, relying on real input from the instance context (for example, the emergency department, the livestock barn) for the development and validation of the model parameters. In this way, ABM development provides a natural opportunity for educational interactions among engineering disciplines, and between engineering and other disciplines.

Overall, this interdisciplinary opportunity is demonstrated in ABMs' reliance on data. The development of the parameters that define the underlying topography and the agents types, characteristics of each type, behaviours, and interactions all rely on accurate and representative data. Opportunities related to data include sourcing and generating data (e.g. data mining, on-site data collection), handling (e.g. extracting, fusing), analysis (e.g. statistical manipulation, inferencing, generalizability), and validation.

The data inputs into an ABM are both technical (in the broad sense, of being specific and precise) as well as social, and can often be gained from non-obvious and disparate sources. In all cases, though, the inputs are inter-disciplinary. Census data, community planning reports, revenue services, and other inputs from the public policy realm are inputs into the community-level epidemic modeling application. In epizootic modeling, context-specific data from veterinary sciences as well as ILPO personnel are required, in order to build a model that accurately reflects the agents' characteristics and behaviours. For any ABM applications within healthcare settings (whether related to patient access issues, infection spread modeling, or other issues), the input of a wide range of healthcare practitioners is required.

Beyond the technical inputs of the respective disciplines in characterizing agents and their behaviours, there is a significant sociological input required for accurate modeling, primarily at the level of agent behaviours and interactions. This can be related to the 'culture' of the modeling context, whether it is an institutional context (the organizational culture) or at the community-level context. In considering the ABM application for community-level epidemic modeling, the model requires sociological input into people's actions in stress and crisis. For example, it has been noted that a strong sense of community is often a defining feature in successful mitigation and recovery in a disaster [36]. This is one example of specific sociological phenomena that directs agent behaviour and interactions in an ABM. Panic behaviour is another such phenomenon, which in its traditional interpretation leads to the idea that fear and uncertainty creates chaos (behaviours and interactions that are decidedly less rational and more difficult to infer), and has a high potential to exacerbate a disaster. As such, these are critical ABM inputs that require a significant amount of attention for accurate characterization. Yet, within an ABM, the sociological dynamics are also directly correlated to physical inputs, such as closing major traffic arterials and bridges to bound agent movement in the community-wide epidemic model. These applications provide students with an opportunity to explore the interactions of the sociological issues within the technical context.

The technical data set also provides interdisciplinary educational opportunities. While there is an explicit opportunity to refine the sensitivity, precision, and accuracy of the models (inherently a computer engineering challenge), the opportunities for direct data generation also present opportunities within engineering and other disciplines. Where agent movement or flow is a parameter of interest – which is the case in all of the applications described – there are opportunities to generate data from existing and new sources. When generating data from existing sources, these sources are often non-obvious, in that they were initially derived for other purposes. Examples relevant to the community-level epidemic model include tracking agents through extracted MAC addresses snooped from commercial electronic devices, cellular location services, street-level surveillance cameras, and arterial traffic monitoring of flows in and out of a city. An example relevant to the patient wait time in healthcare application is to gather sanitized agent data from medical record systems. In many applications, the opportunities exist to develop and implement Real Time Location System (RTLS) applications, such as RFID, to generate location-specific data on agent movement, whether those be individuals in a community, animals in an ILPO, or patients in a hospital.

Where applications involve modeling of disease spread, the validity and reliability of the models will depend on the accurate epidemiological inputs related to (but not limited to) the disease's incubation period, infection period, infection probability, disease duration, virility, morbidity, and mortality probabilities. This is an explicit interdisciplinary input, whether from veterinary medicine or human medicine. It immediately follows that knowledge translation between fields requires a concerted development of communication modalities across disciplines, to receive and give information in ways that maintain the integrity of the data and comprehensible for all parties.

While these represent a range of quantitative data sources, specific and detailed data can also be generated through qualitative methods, such as focus groups and structured interviews with experts in the respective areas. These experts could be emergency measures officials outlining the details of

community-level disaster plans, technicians working in ILPOs providing input into animal management and animal care practices, or physicians and nurses advising on the factors that influence patient movement and care in an emergency department. This vital contribution to ABMs adds context, nuance, and realism to data derived from other sources, and provides students with opportunities for cross-training in qualitative research methodologies more prevalent in the social sciences. This process not only generates data but is also a critical means of validating the model through representative 'expert agents'.

The qualitative focus is evident in other aspects of ABM development as well. ABMs start as coarse-grained models and move toward increased states of refinement. Their utility in early stages of development lie in their ability to provide qualitative insights into the *relative* sensitivities and impacts of varying the model parameters, to illuminate scenarios worthy of more complex investigation, and to iteratively validate the models as they continue to be extended. At these later stages of development and refinement, accurate and precise system metrics amenable to statistical processing become reasonable model outputs. The development process – from qualitative to quantitative – offers students an opportunity to appreciate the scope and limitations of each output.

While certainly not unique as an opportunity within computer engineering, this discussion has highlighted the ways in which the development and refinement of ABM applications engage students with technical and social concepts beyond computer engineering, team students with content area experts in other fields, and by doing so, develop a range of professional skills.

V. DISCUSSION AND CONCLUSION

The highly integrative nature of ABM development is supported by strong teaching and learning theory, where learning is conceived in three major domains: the cognitive domain, the skill or competence domain, and the value or affective domain. Within these three domains, students move from ignorance to knowledge, from inability to competence, and from indifference to understanding, respectively [37]. Development in the cognitive domain can be conceptualized as the development of cognitive structures, which are mental structures of organized knowledge [38] – or, a 'mind map'. Cognitive structures are developed by presenting materials in meaningful ("real-world") contexts, in terms of the relevance and applicability of the materials. Cognitive structures are further developed by requiring students to actively organize materials, assimilating and organizing new knowledge relative to existing knowledge. The opportunities in ABM facilitate these aspects of cognitive development.

Development in the skill domain is more self-evident: students are afforded opportunities to improve their programming skills and data collection, handling, and analysis skills – with all the concomitant possibilities afforded in hardware development and software manipulation. The interdisciplinary interactions of ABM applications also foster a degree of students' development in the affective or value domain, which tends to move through distinct phases in students' undergraduate years. In general, students' value positions tend to move from simplicity and absolutism to complexity and relativism, from concreteness to abstractions, and from external to internal regulation of behaviours [39]. The need to negotiate between different interdisciplinary perspectives, the critical analysis required in model-building, the need to choose best-fits from among multiple possibilities, and the flexibility of ABMs to address broader societal issues are just a few examples of the aspects of ABM development that foster students' development in the value domain.

At a superficial level, the input parameters into an ABM are very context-specific for a particular application: an emergency department, an animal barn, etc. However, at a fundamental level, the ABM shares conceptual similarities with approaches in other fields. Although the descriptive language may vary from one field to the other, the underlying mental constructs are commensurate to one another. For example, many electrical and computer engineers are very familiar with network simulation tools. A University-based example is the tools associated with the ns-2 simulation platform. In this environment, models of telecommunication systems are built, with many modeling variants of Internets, etc.

In some ways, the particular application area of an ABM is peripheral to the ABM's inherent ability to foster the mental constructs that characterize the underlying computer engineering epistemology. Epistemology encompasses the nature of knowledge, methods of acquiring knowledge, and the limits of knowledge in a field, as well as the mental models of understanding the world. Among other components, the epistemology of a field encompasses its defining problems (problems of concern), its fundamental concepts (operational principles, normal configurations), and its theoretical tools (its methods, theories, and the language to express these) [40].

This paper has presented agent-based modeling toward several diverse applications, as a means for computer engineers to learn and integrate concepts in sociology, psychology, public policy, health, and business, among others. This supports the call – across engineering disciplines – for broader educational exposure to non-engineering areas that ultimately foster social awareness and ethical consciousness in a globalized society, and contribute to a skill set that ultimately fosters engineers' leadership abilities in the public realm.

REFERENCES

[1] ABET Inc., *Criteria for Accrediting Engineering Programs effective for Evaluations during the 2009-2010 Accreditation Cycle.* Baltimore, MD: ABET Inc., 2008.

[2] IEEE Computer Society, *Computing Curriculum 2001, Computer Science: Final Report* [Online]. Available: www.ieee.org.

[3] National Academy of Engineering, *The Engineer of 2020: Visions of Engineering in the New Century.* Washington, DC: National Academies Press, 2004.

[4] R.D. Schuler, "Interdisciplinary centers: A natural and necessary force for creativity and change in engineering

research and education," *J. Eng. Education,* vol. 83, no. 1, January 1994.

[5] J. Trevelyan, "Technical coordination in engineering practice," *J. Eng. Education,* vol. 96, no. 3, pp. 191-204, July 2007.

[6] American Society of Civil Engineers, *Civil Engineering Body of Knowledge for the 21st Century*, 2004 [Online]. Available: www.asce.org.

[7] American Society of Mechanical Engineers Council on Education, *A Vision of the Future of Mechanical Engineering Education*, 2004 [Online]. Available: http://files.asme.org/asmeorg/Education/College/ME/7782.pdf.

[8] M. Krishnan, M.J. Paulik, S. Yost, and T. Stoltz, "Shared projects with a multi-sub-disciplinary flavor – providing integration and context in a new ECE spiral curriculum," in *38th Annual Frontiers in Education Conference,* Saratoga Springs, New York, 2008.

[9] A.B. Albu, K. Malakuti, H. Tuokko, W. Lindstrom-Forneri, and K. Kowalski, "Interdisciplinary project-based learning in ergonomics for software engineers: A case study," in *Third International Conference on Software Engineering Advances,* Sliema, Malta, 2008.

[10] S.H. Bhavnani and M.D. Aldridge, "Teamwork across disciplinary borders: A bridge between college and the work place," *J. Eng. Education*, vol. 89, no. 1, January 2000.

[11] R. Pool, *Beyond Engineering: How Society Shapes Technology.* Oxford, England: Oxford University Press, 1997.

[12] National Engineering Summit 2009, Montreal, Quebec, Canada, www.engineeringsummit.ca

[13] H. Andersson and T. Britton, "Stochastic epidemic models and their statistical analysis," Lecture Notes in Statistics, Vol. 151, ISBN: 978-0-387-95050-1, 2000.

[14] F. Carrat, J. Luong, H. Lao, A. Sallé, C. Lajaunie, and H. Wackernage. (2006). A 'small-world-like' model for comparing interventions aimed at preventing and controlling influenza pandemics. *BMC Medicine* [Online]. Available: http://www.biomedcentral.com/1741-7015/4/26.

[15] MITACS (2009). Transmission dynamics and spatial spread of infectious diseases: Modeling, prediction and control. [Online]. Available: http://www.mitacs.ca/main.php?mid=10000015&pid=75&proid=28

[16] E. Bonabeau, (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Science* [Online]. *99(Suppl 3),* pp. 7280-7287. Available: http://www.pnas.org/content/99/suppl.3/7280.full#xref-ref-3-1

[17] J.M. Epstein and R.L. Axtell, *Growing Artificial Societies: Social Science from the Bottom Up.* Cambridge, MA: MIT Press, 1996.

[18] R. Axelrod, *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton, NJ: Princeton University Press, 1997.

[19] J.M. Epstein, "Artificial society: Getting clues on how a pandemic might happen by creating a huge model of the United States," The Brookings Institution. [Online]. Available: www.brookings.edu/interviews/2008/0402_agent_based_epstein.aspx.

[20] National Institute of General Medical Sciences, "MIDAS: Models for infectious disease agent study". [Online]. Available: https://www.epimodels.org/midas/about.do

[21] M. Borkowski, B.W. Podaima and R.D. McLeod, "Epidemic modeling with discrete space scheduled walkers: Possible extensions to HIV/AIDS," *BMC Public Health*, to be published.

[22] K.D. Rossmo, *Geographic Profiling*. Boca Raton, FL: CRC Press, 1999.

[23] CDC (2008). Prevention and control of influenza recommendations of the Advisory Committee on Immunization Practices (ACIP), 2008. [Online]. Available: http://www.cdc.gov/mmwr/PDF/rr/rr57e717.pdf

[24] B. Paizen, M.R. Friesen, J. Kraut, and R.D. McLeod, "Epizootic agent based modeling: The mobility threshold for disease spread as a critical percolation phenomenon," submitted for publication.

[25] S. Mukhi and M. Laskowski, "Agent-based simulation of emergency departments with patient diversion," presented at eHealth2008, London, UK, September 8-9, 2008.

[26] M. Laskowski, R.D. McLeod, M.R. Friesen, B.W. Podaima, and A.S. Alfa, "Models of emergency departments for reducing patient waiting times," *PlosOne*, to be published.

[27] L. Patvivatsiri. (2008). A simulation model for bioterrorism preparedness in an emergency room [Online]. Available: http://www.informs-sim.org/wsc06papers/061.pdf.

[28] W. Bhanzhaf, P. Nordin, R. Keller, and F. Francone, *Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and its Applications.* San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1998.

[29] M. Laskowski and S. McGrath, "Effects of lying in reputation-based multi-agent systems," in *Elect. and Comput.Eng.2005: Canadian Conf. on*, 2005, pp. 1014-1018.

[30] D. Sanders, S. Mukhi, M. Laskowski, M. Khan, B.W. Podaima, and R.D. McLeod, "A network-enabled platform for reducing hospital emergency room waiting times using an RFID proximity location system," presented at 19th Int. Conf. on Syst. Eng., Las Vegas, NV, 2008.

[31] R.D. McLeod, M. Laskowski, M.R. Friesen and B.W. Podaima, "Agent based models for nosocomial infections: Modeling of hospital-acquired infections within a hospital," Report for the Public Health Agency of Canada, available from authors, 2009.

[32] C. van den Dool, M.J.M. Bonten, E. Hak, J.C.M. Heijne and J. Wallinga. (2008, October). The effects of influenza vaccination of health care workers in nursing homes: Insights from a mathematical model. *PLoS Med* [Online]. *10(5).* Available:

http://medicine.plosjournals.org/archive/1549-1676/5/10/pdf/10.1371_journal.pmed.0050200-L.pdf

[33] B. Bean, B.M. Moore, B. Sterner, L.R. Peterson, D.N. Gerding and H.H. Balfour Jr., "Survival of influenza viruses on environmental surfaces," *J. Infectious Diseases*, vol. 146, no. 1, pp. 47-51, Jul 1982.

[34] Y. Thomas, G. Vogel, W. Wunderli, P. Suter, M. Witschi, D. Koch, C. Tapparel and L. Kaiser, "Survival of influenza virus on banknotes," *Appl. and Env. Microbiology,* vol. 74, no. 10, pp. 3002-3007, May 2008.

[35] CDC SARS Fact Sheet http://www.cdc.gov/ncidod/sars/pdf/factsheet.pdf

[36] D. Paton and D. Johnston, "Disasters and communities: vulnerability, resilience and preparedness," *Disaster Prevention and Manage.: An Int. J.,* vol. 10, no. 4, pp. 270-277, 2001.

[37] C. Fincher, "Learning theory and research," in *Teaching and Learning in the College Classroom,* 2nd ed., K.A. Feldman and M.B. Paulsen, Eds. Needham Heights, MA: Simon & Schuster Custom Publishing, 1998, pp. 57-80.

[38] W.J. McKeachie, P.R. Pintrich, Y. Lin, D.A.F. Smith, and R. Sharma, "from: Teaching and learning in the college classroom: a review of the research literature," in *Teaching and Learning in the College Classroom,* 2nd ed., K.A. Feldman and M.B. Paulsen, Eds. Needham Heights, MA: Simon & Schuster Custom Publishing, 1998, pp. 81-115.

[39] J. Kurfiss, "Intellectual, psychosocial, and moral development in college: four major theories," in *Teaching and Learning in the College Classroom,* 2nd ed., K.A. Feldman and M.B. Paulsen, Eds. Needham Heights, MA: Simon & Schuster Custom Publishing, 1998, pp. 139-161.

[40] W.G. Vincenti, *What Engineers Know and How They Know It*. Baltimore, MD: Johns Hopkins University Press, 1990.