VALIDATION AND EVALUATION OF EMERGENCY RESPONSE PLANS THROUGH AGENT-BASED MODELING AND SIMULATION

Joseph Edward Helsing

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

May 2018

APPROVED:

Armin R. Mikler, Major Professor
Rodney Nielson, Committee Member
Bill Buckles, Committee Member
Chetan Tiwari, Committee Member
Barrett Bryant, Chair of the Department of Computer Science and Engineering
Costas Tsatsoulis, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Helsing, Joseph Edward. *Validation and Evaluation of Emergency Response Plans through Agent-Based Modeling and Simulation*. Doctor of Philosophy (Computer Science and Engineering), May 2018, 103 pp., 12 tables, 54 figures, 66 numbered references.

Biological emergency response planning plays a critical role in protecting the public from possible devastating results of sudden disease outbreaks. These plans describe the distribution of medical countermeasures across a region using limited resources within a restricted time window. Thus, the ability to determine that such a plan will be feasible, i.e. successfully provide service to affected populations within the time limit, is crucial. Many of the current efforts to validate plans are in the form of live drills and training, but those may not test plan activation at the appropriate scale or with sufficient numbers of participants. Thus, this necessitates the use of computational resources to aid emergency managers and planners in developing and evaluating plans before they must be used. Current emergency response plan generation software packages such as RE-PLAN or RealOpt, provide rate-based validation analyses. However, these types of analysis may neglect details of real-world traffic dynamics. Therefore, this dissertation presents Validating Emergency Response Plan Execution Through Simulation (VERPETS), a novel, computational system for the agent-based simulation of biological emergency response plan activation. This system converts raw road network, population distribution, and emergency response plan data into a format suitable for simulation, and then performs these simulations using SUMO, or Simulations of Urban Mobility, to simulate realistic traffic dynamics. Additionally, high performance computing methodologies were utilized to decrease agent load on simulations and improve performance. Further strategies, such as use of agent scaling and a time limit on simulation execution, were also examined. Experimental results indicate that the

time to plan completion, i.e. the time when all individuals of the population have received medication, determined by VERPETS aligned well with current alternate methodologies. It was determined that the dynamic of traffic congestion at the POD itself was one of the major factors affecting the completion time of the plan, and thus allowed for more rapid calculations of plan completion time. Thus, this system provides not only a novel methodology to validate emergency response plans, but also a validation of other current strategies of emergency response plan validation.

ACKNOWLEDGMENTS

This work would not have been possible without the support of the University of North Texas's Computer Science and Engineering Department and the Library and Information Science Department, both of which have provided me with not only a host of mentors and peers to work with and learn from, but also the financial support to complete this work. I am particularly grateful to my committee chair, Dr. Armin Mikler for his mentoring and unending support in my pursuit of this degree.

I would like to thank the current and previous members of the CERL Lab (Faris, Cree, Tanah, Marty, Richard, Sampson, Pat, Joe, Sarat, Josh, Nirosha, Kaytlin, Clayton, and the rest) whose continued aid, suggestions, and brutally honest critiques of my work have enabled me to achieve this goal. I will be sure to carry forward our candid methods of improving writings and presentations to future teams and pupils. It has been a honor to work with you all and call you my fiends and labmates.

I would also like to thank my fellow graduate student peers, Kate Lester, Ryan Michaels, and Brenda Cantu, for their time spent conversing about our respective research and providing kind words and stout advice. Their unique perspectives and backgrounds have provided numerous insights and revelations during my time at UNT. The importance of friendship and family cannot be understated in the process of obtaining a PhD.

Most importantly of all, I would like to thank my family. My Mom and Dad, Judith and Edward Helsing have been an unwavering rock of emotional, and financial, support during my education. They have consistently pushed me to excel and achieve throughout my life and I cannot thank them enough for helping to become the person I am today. I love you both dearly and look forward to more time spent together in the future. I would also like to thank my girlfriend, Harsha Gwalani, whose kind words, endless encouragement, and boundless love have helped me through the completion of my dissertation. I am excited to return the favor and start on a new chapter together.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Biological emergency response planning plays a crucial role in protecting the public from possible devastation resulting from disease outbreaks, be they naturally occurring or man-made. A response plan's utility extends only as far as its ability to service affected populations within a time limit during an emergency. Thus, validating that a plan will succeed when executed is just as important, if not more so, than actually generating the plan. This dissertation seeks to address challenges faced when attempting to computationally validate biological emergency response efficiently and effectively.

1.1. Motivation

In 2001, not long after the events of September 11th, several lethal letters were mailed through the United States postal system to five major news agencies and two U.S. senators. Some of these missives contained a highly concentrated dose of almost pure anthrax spores, which led to multiple individuals becoming infected when they inhaled the spores. These attacks led to the death of five individuals and caused 17 others to contract the disease, in what the Federal Bureau of Investigation dubbed the Amerithrax case and referred to as "the worst biological attacks in U.S. history." Though it was later discovered that these attacks were carried out by Dr. Bruce Ivins, a biodefense researcher working for the United States Army Medical Research Institute of Infectious Diseases, the attacks prompted a shift in how the United States government responds to biological emergencies [21].

Specifically, the United States Congress passed a series of acts which began a number of initiatives to better protect US citizens from biological emergencies, be they natural or man-made. One of the first, post-September 11th acts to primarily focus on preparedness through medical countermeasures was the Project BioShield Act. This act called for the allocation of roughly five billion dollars towards the purchasing of vaccines for the public in the event of an attack using biological, chemical, radiological, or nuclear agents. Further, it reassigned management of the Strategic National Stockpile (SNS), where such medical

countermeasures would be stored, to the Department of Health and Human Services (HHS) [62, 1].

That same year, the Cities Readiness Initiative (CRI) was established at the Centers for Disease Control and Prevention (CDC). Considering that more than 50% of the country's population resides in major cities, the CRI was specifically targeting the nations largest population centers to enhance preparedness in the event of a biological disaster. The CRI was designed to coordinate the use and distribution of necessary medical supplies and countermeasures from the SNS to the affected public. The scope of the CRI was expanded in 2013, through the Pandemic and All-Hazards Preparedness Reauthorization Act, to include natural and man-made public health threats [50].

In order for cities to adequately prepare for potential biological emergencies, two key steps need to be taken. First, response plans need to be developed for how a city would actually distribute medical countermeasures. This task is often performed by emergency managers and planners, who use personal knowledge of the area to decide where and how points of dispensing (PODs) should be set up to distribute these countermeasures in the event of an emergency. More recently, emergency managers and planners are utilizing software systems, such as RE-PLAN [51] or RealOpt [35], to gain more information about the area and make decisions with the aid of computer algorithms. Furthermore, the CDC and the Department of Health and Human Services (DHHS) have provided a list of POD standards that ensure POD allocation, placement, and layout are all meeting an acceptable level of service to the public [39]. A list of POD standards can be found in Table A.1.

Second, the execution of these plans needs to be evaluated and validated, which is typically achieved through the use of drills and exercises. For example, the New York City Health Department (NYCHD) operates the Rapid Activation for Mass Prophylaxis Exercise (RAMPEx) to test the ability of its agencies and workers to rapidly mobilize its mass distribution of medical countermeasures systems. With little notice, emergency response workers and agencies must set up 30 PODs and several command and control centers quickly to simulate real-world activation in the case of an actual emergency. The execution of this

plan is then evaluated by NYCHD, and alterations are made as necessary [64, 41]. Many other cities, counties, and states across the country also engage in similar drills, such as Peoria, Illinois [2], Houston, Texas [54], Yolo County, California [66], and the entire State of Louisiana [49].

However, despite these efforts, live testing of plan execution still poses great challenges for emergency managers. A 2012 study, produced by the RAND Corporation at the request of the CDC, examined whether communities involved in the CRI were able to meet the 48 hour window for vaccinating their respective areas. While the study found that many communities scored well in their abilities to vaccinate the public, it was determined that testing for plan execution was not being conducted at an appropriate scale. Of the 1,422 drills executed during 2009 and 2010, 90% involved 100 or fewer civilians, and 68% of drills that utilized PODs involved 500 or fewer civilians. This may be due to a variety of reasons, such as funding and the difficulty in organizing a multitude of staff and mock patients [40].

These challenges with live testing pose a significant risk to the public at large; therefore it is critical to find alternatives to exercises and drills as a means for evaluating and validating plans. Specifically, both the feasibility and efficacy of a plan must be thoroughly analyzed before that plan is executed. Feasibility refers to a plan's ability to meet certain key criteria, such as being able to medicate the entirety of a population within a specified time window. If a plan is determined to be feasible after testing, it can be deemed usable for a given emergency scenario. Efficacy describes, in general, how well a plan performs during ideal conditions or a controlled scenario, such as a simulation. This can be measured in terms of how POD placement or the removal of segments of the road network affects network wide traffic flow. Information of that nature can help distinguish between similarly feasible plans, in order to select a more optimal plan for later execution.

Despite tools and techniques being available for the design and basic analysis of biological emergency response plans, gaps remain in the validation and evaluation of these plans. First, while these tools and techniques have been developed to provide emergency managers and response planners the ability to test how the execution of a plan might function,

they have either done so with insufficient realism, or at an inadequate scale. Second, it is necessary that there be a discussion of the theoretical aspects of validating a response plan, such as determining which methodologies are appropriate for certain situations and to what extent should realism be a driving factor in modeling and simulating a response plan execution. My dissertation seeks to fill these gaps in the greater body of work on biological emergency response plan validation and evaluation.

1.2. Contributions

This dissertation addresses the following topics: biological emergency response plan execution simulation; biological emergency response plan validation and evaluation. Following is a summary of how these topics will be incorporated into this research and what will be contributed to each.

1.2.1. Biological Emergency Response Plan Execution Simulation

Much of the current research in biological emergency response plan execution, as described previously and which will be expanded upon in the literature review, is carried out through the use of live drills and exercises. While the scale may be appropriate, as with some of New York City's drills [64, 41], it does not include the entire population of the area, and most places lack the resources and coordination to manage this type of drill. More importantly, attempting to test new strategies can be highly resource-intensive with respect to time and money. This research seeks to explore alternatives to traditional live methods of emergency response plan activation to test new scenarios. Specifically, this research seeks to utilize the mezzo-scale dynamics that appear when using an agent-based simulation. By simulating at this scale, solely relying on rates of traffic flow across the network can be exchanged to focus on behaviors that emerge from utilizing individual vehicles.

1.2.2. Emergency Response Plan Validation and Evaluation

There exists previous research in this area for the evaluation of response plans. As stated earlier, the CDC has a set of POD standards that should be adhered to [24], as well as the Medical Countermeasure (MCM) Operational Readiness Review (ORR) to eva-

luate whether an area has formed a plan that meets those standards [26]. However, while these standards are vital for ensuring response plans exist and function properly during an emergency, they lack nuance and the ability to evaluate the degree of feasibility or efficacy.

Additionally, while some computational systems have been developed to aid in the evaluation and validation of response plans, such as RE-PLAN [51], RealOpt [36], iERF [32, 31], there has still yet to be an in-depth analysis of how to validate a response plan. This research seeks to provide that analysis, by examining how techniques from modeling and simulation validation research can be applied to the features of a response plan, such as evaluating the level of realism retained by the abstraction of that response plan into a computational format and determining if the simulated processes are reflective of real processes.

1.3. Research Questions

The following questions will be the focus of this research:

(1) How can agent-based simulations be used to validate emergency response plans?

(2) How do agent-based simulations of emergency response plan activation compare with analytical examinations?

The remainder of this dissertation will divided into the following chapters. Chapter 2 provides background information on the field of emergency management and a literature review of current work in emergency management simulation and modeling. It will also review the utility of simulation for evaluation and study in different contexts. Chapter 3 discusses simulation validation in general and then several metrics that can be used to evaluate emergency response plans. It continues by describing the VERPETS framework, its data flow, and the functionality it provides to emergency planners. Chapter 4 highlights the experiments designed to show that effective agent-based validation can be achieved through computational means. Chapter 5 presents and discusses the results of the aforementioned experiments, along with a summary of this dissertation.

CHAPTER 2

BACKGROUND

In order to provide a better understanding of how this research applies to the fields of emergency management, this chapter will provide background information intertwined with an examination of the relevant literature. It will start with a general discussion of the field of emergency management, which will transition into a more in-depth discussion of biological emergency response plan evaluation. Finally, there will be a discussion of modeling simulation in multiple contexts.

2.1. Emergency Management

The field of modern emergency management, also known as disaster management, focuses primarily on the four phases of the emergency management cycle: Mitigation, Preparedness, Response, and Recovery. Figure 2.1, courtesy of FEMA, shows the sequence of phases. The emergency management cycle can begin in any of the four phases, though, typically emergency management begins with either the mitigation phase or preparedness phase before an emergency occurs or the response phase during an emergency. However, since many areas of the US have not experienced a man-made biological attack, while still being mandated to prepare for such an event, this discussion will begin with the mitigation phase. This phase focuses on "...preventing future emergencies or minimizing their effects..." [19] and occurs after one disaster and before the next. The focus of this phase is typically prevention of emergencies, and thus is typically associated with activities that either help reduce the chances of an emergency occurring, or decrease the damage caused in the case of unavoidable emergencies. Examples might be the installation or reinforcement of levees to help prevent floods, the improvement of building codes to reduce the damage caused by earthquakes [22], or the use of a yearly flu shot to help reduce a populations risk of an outbreak occurring.

While similar to mitigation, preparedness concentrates on "...preparing to handle an emergency...," [19] and occurs only before an emergency develops. This preparedness

FIGURE 2.1. The emergency management cycle.

can take many forms such as generating necessary supply lists for potentially affected populations [46], stockpiling medicines and prophylactics [23], or training, organizing, and exercising responses to emergencies [61]. Another key facet of preparedness is the formation of emergency response plans. These plans are designed to guide responders and civilian populations during the course of an actual emergency, and are critical to successfully mounting a response effort. Also important to note, each emergency response plan must be tailored to the emergency, area, scope, and population that it is attempting to aid to best mitigate the emergency. Validating these plans will be the focus of this dissertation and will be discussed in greater depth later.

Following preparedness, the response phase of the cycle occurs only during an emergency. This is when preparations made during the mitigation and preparedness phase are utilized. It should be noted that if the preparedness phase is executed correctly, this should not be the first time a plan is examined for potential problems [19]. Examples of actions during the response phase include opening up contraflow lanes on highways during a mass evacuation before a hurricane, the strategic deployment of PODs to supply a population with

prophylactics in the event of a disease outbreak, or following set guidelines in the handling of Ebola patients [10]. These are all measures which need to have been prepared in advance to better organize their execution during a response effort.

Finally, after the emergency has occurred, emergency managers and planners move into the recovery phase. In this phase, managers and responders attempt to "... return to a normal or an even safer situation following an emergency [19]." This includes supplying basic necessities such as food and water to affected individuals and families, providing mental health services to those who were severely impacted by the disaster, administering first aid, and providing basic cleaning and health supplies to the affected [4]. Additionally, this includes helping affected individuals and families receive financial assistance to offset or cover the cost of repairs to damaged homes and property. This phase will then flow into the mitigation phase, where emergency managers again take stock of the problems that occurred during the disaster and how they can be mitigated in preparation for the next emergency [19].

2.2. Emergency Response Plan

As previously stated, emergency response plans play a critical role in providing emergency responders, and the general populace, with a guide for how to act in given situations. These can be created for a variety of situations and some of the key features of a plan are the type of emergency, the area affected, the scope, and the population that it is attempting to aid. They can be as common place and straightforward as a fire escape plan for a building, as seen in Figure 2.2, or as complicated as a statewide evacuation from an impending hurricane, as seen in Figure 2.3. For example, due to the structural devastation that can be caused by a hurricane, in terms of both high sustained wind speeds, prolonged flooding, and the storm surge, the most effective means of protecting the public is via evacuation. To aid in this evacuation, states along the coast of the Gulf of Mexico, such as Louisiana, have created supply lists to aid the public in knowing what supplies to gather in preparation [46] and have constructed contraflow lanes on major highways to improve traffic flow out of the cities. This requires coordination of the general populace and police and emergency services

to evacuate people in time.

FIGURE 2.2. An example fire escape plan for a hotel [18].

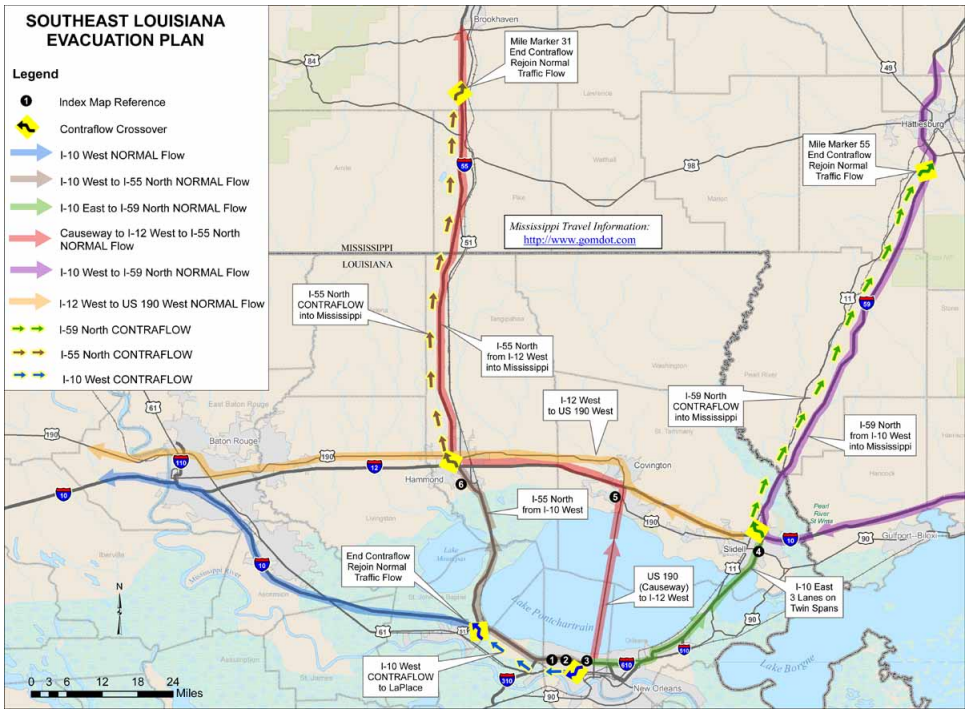


FIGURE 2.3. An evacuation plan for southeast Louisiana [47].

In terms of emergency response plans for a disease-based emergency, which is the focus of this research, there are multiple strategies. One of the oldest and most widely used to prevent potential contamination has been quarantining. Deriving from the Italian word "quaranta", meaning 40, quarantines were used during the 14th century to attempt to contain and prevent the spread of plague. Those that were determined to be ill could

be transported to plague hospitals, known as lazarettos. These institutions were frequently separated from the city by a natural water barrier, when available, or moats or ditches filled with water when no natural barrier could be used. At ports, those individuals suspected of carrying plague would be isolated in buildings for 40 days until they were deemed no longer a threat. Additionally, some city-states would either bar entry to merchants or strangers suspected of carry the disease, or created a sanitary cordon to separate the sick and healthy. So great was the risk to the populous, many of these response plans were enforced under penalty of death [59].

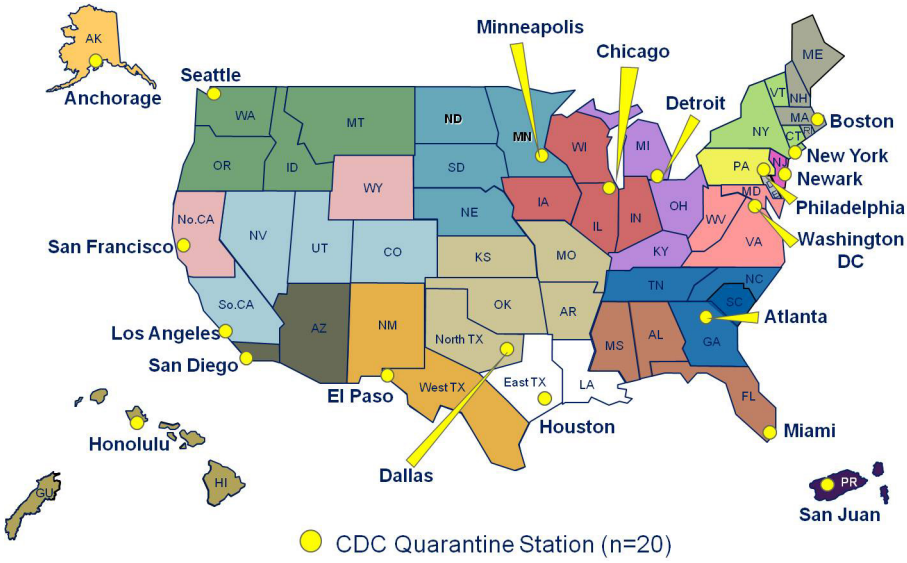


FIGURE 2.4. A map of DGMQ locations around the United States [9].

Even today quarantining is still a popular method for preventing disease spread. In the U.S., The Division of Global Migration and Quarantine (DGMQ), managed the CDC, has specific response plans in place on how to respond to threats. Set up at major travel hubs around the U.S., as seen in Figure 2.4, the DGMQ is constantly surveying passengers and goods that enter the country. Certain animals, such as monkeys, bats, and rodents, are closely monitored for disease and may be confiscated if they test positive. Additionally, individuals may be barred from travel if they diagnosed with a particular illness, such as Ebola [9]. Further, plans are in place for how to respond to suspicion that a passenger on a flight may have exposed others to diseases such as measles, pertussis, or tuberculosis, by

contacting and testing those potentially exposed [25]. An example seating chart of who to contact in the event of possible exposure to certain air-borne infectious diseases can be seen in Figure 2.5.

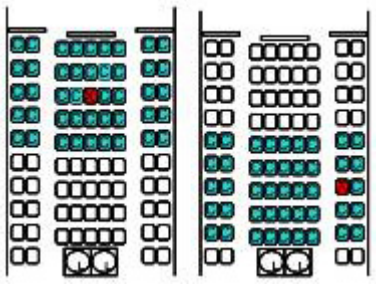


FIGURE 2.5. Seating diagram for passengers exposed to measles, rubella, or TB. The red seat indicates the infected passenger. [25].

Another strategy, which will be the focus of this dissertation, is that of a POD-based response plan. In the case of the release of anthrax, the affected population needs to receive medical countermeasures, such as Raxibacumab®, Anthrasil®, or BioThrax®[45], within a relatively small time window. In order to insure that the response effort has the best possible chance of succeeding an emergency response plan that details POD locations, and which portions of the population are assigned to which POD must be generated. In order to construct this plan, a response planner will start by assessing what resources, such as available staffing and supplies to set up booths to distribute medical countermeasures, are currently available within the area to construct and manage PODs. According to CDCs standards for PODs, the number of PODs must meet the limit described in Equation 2.1 [24]. Because a fixed processing throughput is assumed at each POD, and a limited time window is allowed, there must be an adequate number of locations available to process the total population. The time window is based on the original, CDC mandated 48 hour window minus the "...CDC's assumption that it might take up to 12 hours for initial delivery of [material]..." and the "...Target Capabilities List's assumption that it might take up to 12 hours to get [material] from warehouses to PODs [24]." While that would normally only leave responder with 24 hours with which to medicate the public, based on assumptions from regional planners this window can be expanded to 36 hours. This is due to the ability

for some of the steps in both the delivery of medication and setting up of PODs to be done concurrently. Further, this window may be modified by planners using their knowledge of the region and local area. In addition to the time window, POD throughput must be determined. This is accomplished through timing drills and exercises, which double as training experiences for the staff and volunteers.

$$Number\ of\ PODs \geq \frac{Population\ visiting\ PODs\ in\ person}{Hourly\ per\ POD\ throughput * 36\ hours} \tag{2.1}$$

Based on this information, the planner will determine how many PODs must be set up, and then investigate potential locations for those PODs within the area, while determining how to allot the population amongst those PODS. Most potential locations are government owned, such as public schools and stadiums, and in some cases hospitals are also used. Finally, the response plan needs to be evaluated against the POD standards, as seen in Table A.1, and further exercises and drills can be performed [41]. Additionally, computational methods may be used to partially validate its feasibility and efficacy [51, 36].

As previously mentioned, in order to create an emergency response plan several pieces of data are needed. First, the geographic distribution of the population is required. The U.S. Census Bureau has divided every county in the U.S. into a hierarchy of different sized geographic spatial units (GSU), as seen in Figure 2.6 with some based purely on geographic area and others based on a population cap per GSU. As shown in the figure, the census block provides the finest granularity of spatial partitioning. Census blocks are typically divided by features such as streets, railroad tracks, rivers and streams, property lines, and city and county boundaries. They are also the smallest unit for which the Census Bureau gathers 100-percent data, i.e. collecting data for each house. Consequently, this sometimes means that some data may be suppressed for certain blocks due to their population being so small that it may lead to individuals in those blocks being personally identified by their census data.

At the next level are block groups, which represent collections of census blocks. Spe-

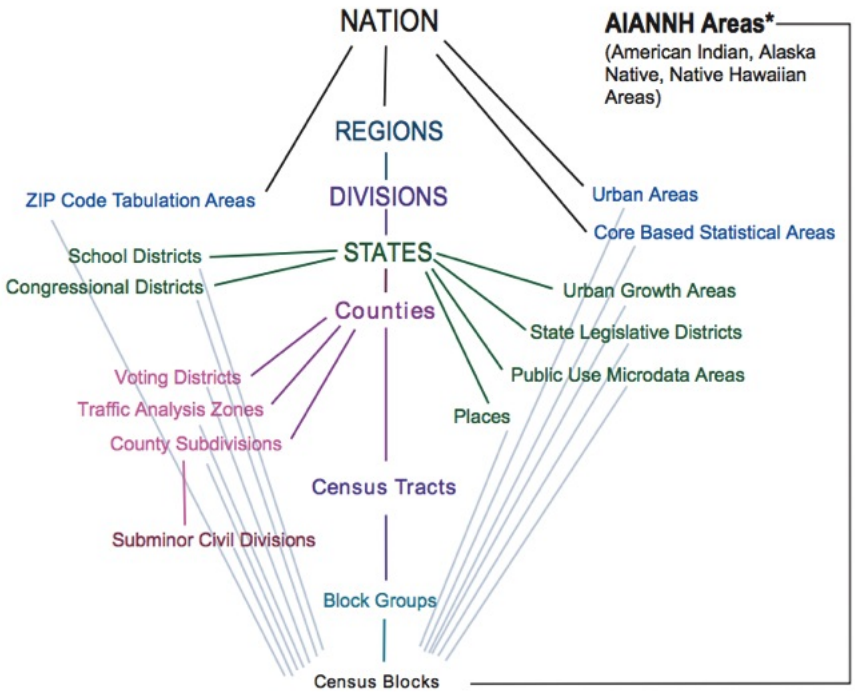
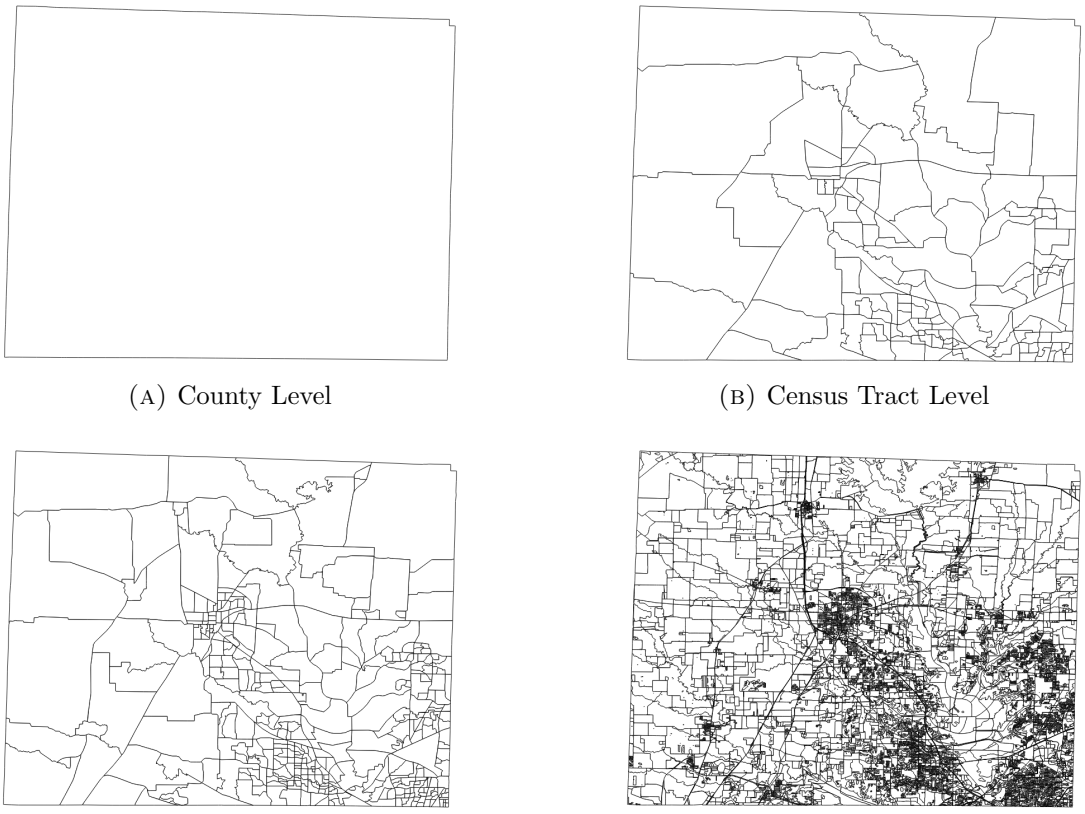


FIGURE 2.6. A hierarchical relationship of geospatial entities according to the U.S. Census Bureau [14].

cifically, a block group is made up of all blocks in a census tract that have the same starting digit in their ID number. By aggregating census blocks, this increases the grouped population and helps to anonymize individuals. Thus, data is less frequently suppressed at this level. Further, they are the smallest GSU that the Census Bureau publishes sample data for, which means data is only collected from a fraction of households.

Above the census block group is the census tract level. Census tracts consist of multiple census block groups, and represent relatively permanent delineations of area within a county. This aids the Census Bureau in measuring changes within the population from one decennial census to another. Additionally, each tract is designed to be as homogeneous as possible when it comes to population demographics. [48] While tracts may be more stable geographic units and allow for greater anonymity, the granularity when examining population distributions becomes more coarse. Therefore, for this dissertation census block groups provided an effective overall balance between data availability and population distribution granularity.



(A) County Level

(B) Census Tract Level

(C) Census Block Group Level

(D) Census Block Level

FIGURE 2.7. Population partitions at different GSU levels for Denton County, TX.

Figure 2.7 provides an example of how a county can be divided into the previously mentioned GSUs. Figure 2.7a shows the county boundaries of Denton County in Texas. Denton County was selected as an example due to its inclusion as one of the locations used for experiments in this dissertation. Figure 2.7b shows how Denton County is divided into census tracts. As can be seen in the figure, higher densities of smaller census tracts represent higher population densities. Several cities can be seen in this map including Denton (in the middle), Lewisville(in the middle on the bottom), Carrollton (on the bottom right corner), and part of Frisco (on the right side), each of which have high clusters of census tracts. Figure 2.7c makes these urban areas even more obvious, as the map has been divided into census block groups. Finally, Figure 2.7d shows the finest granularity partitioning with census blocks.

In addition to population data, the road network for area must be obtained to deter-

mine where effective locations for PODs might be. OpenStreetMap (OSM) provides a free, comprehensive source of road network data for download online. Their data has been used by organizations like Agilogik GmbH, Wikimedia, and MapQuest for tasks including geocoding, routing, rendering, and planning. The road network data itself contains information such as the type of road (highway, residential, service, etc), the number of lanes, the direction of those lanes the speed limit, the name of the road, and the geographic location [13, 53]. Figure 2.8 shows the OSM road network data for Denton county.

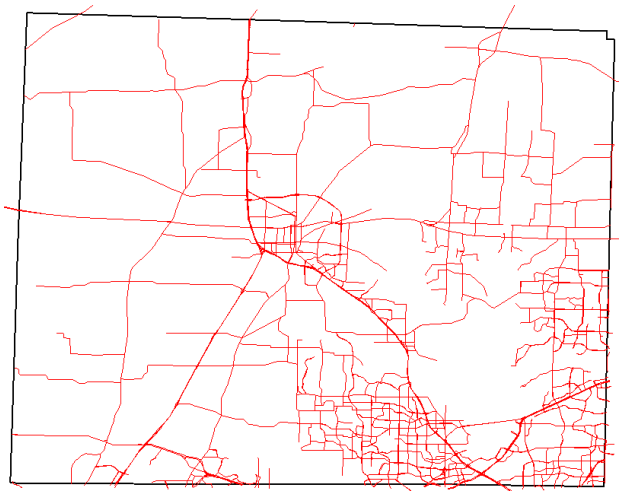
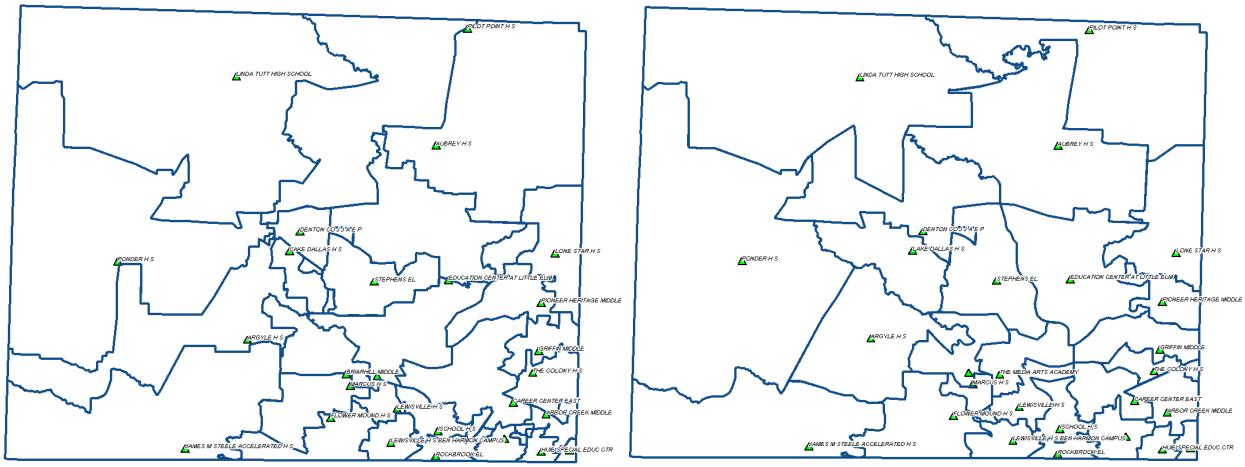


FIGURE 2.8. The road network for Denton County, TX.



(A) Equal Population Partitioning (B) Closest Pod Partitioning

FIGURE 2.9. Two partitionings of the population and subsequent POD assignment for Denton County, TX created via RE-PLAN [51].

One of the final steps in emergency response plan creation is the partitioning of the population into catchment areas. Figure 2.9, shows the final result of an equal population partitioning (EPP) of the census block groups from Figure 2.7c. In this partitioning, all of the census block groups are divided as evenly as possible between the catchment areas. Inside each catchment area is a green triangle representing the POD for that catchment area. Once the PODs have been placed, the GSU's can also be repartitioned into a closest POD partitioning (CPP), to decrease the average travel time for all GSU's in the catchment area. This dissertation will be using emergency response plans with closest pod partitionings.

2.3. Emergency Response Plan Evaluation

In addition to creating emergency response plans, as part of the preparedness phase of the emergency management cycle evaluation and validation of such plans aids in protecting the public if those plans needed to be executed. Further, evaluation and validation is also directly tied to the federal funding which subdivisions of states or a consortium of states receive for emergency preparedness [8]. In order to be eligible for this funding, plans must be generated and adhere to guidelines and meet certain standards such as the Local Technical Assistance Review (TAR) or the Medical Countermeasure Operational Readiness Review (MCM ORR) .

Previously, the CDC and the Division of the Strategic National Stockpile used the TAR to assess if a plan meets national standards. The TAR primarily examined whether the elements it requires were present in the response plan. For instance, one of the first elements on the review is whether or not the local plan "... is presented and includes a medical and health annex, which contains the SNS functional areas of support, and meets NIMS requirements [42]." The next review element examines if the plan is updated annually. Though the review was comprehensive, covering some 84 elements across 12 sections including a tactical communication plan, security, distribution, and training, it lacked the ability to examine nuance and to what degree many of these items functioned [42].

In 2014, the CDC implemented a new method of evaluating POD preparedness and performance, the Medical Countermeasure Operational Readiness Review. This review was

created to better evaluate the ability of a county or state to execute a plan if the need arose. It expanded from the TAR to align with eight public health preparedness capabilities.

- Community Preparedness
- Emergency Operations Coordination
- Emergency Public Information and Warning
- Information Sharing
- Medical Countermeasure Dispensing
- Medical Material Management and Distribution
- Responder Safety and Health
- Volunteer Management

Each of these capabilities is broken into individual functions, which may have multiple parts that represent some specific objective within the capability. Additionally, a function may have a planning implementation and an operational implementation which represents the planning and execution of an emergency response. Further, the MCM ORR transitioned from the TAR scored system to a continuum of levels representing an area's preparedness. In this case, there are four levels representing how many of the individual objectives an area has achieved for a function [26]. For example, the MCM Dispensing capability is broken into five functions:

- Identify and initiate counter measure dispensing
- Receive medical countermeasures at POD
- Activate dispensing modalities
- Dispense medical countermeasures to identified population
- Report adverse events

The "identify and initiate counter measure dispensing" function, further has three parts:

- Guidance/plans document dispensing strategies to include: 1) open public PODs, 2) Close PODs, and 3) Populations with Access and Function Needs
- Guidance/plans document the capability to 1) initiate a dispensing campaign and

2) sustain dispensing campaign follow-on needs

- Guidance/plan identify healthcare partners that would participate in MCM activities and include: 1) list of current healthcare partners with appropriate contact information, 2) MOUs with these organizations, 3) procedures for how these healthcare partners will participate in MCM activities, and 4) planning guidance for those partners participating as closed PODs.

Figure 2.10 shows the planning implementation portion of the "Identify and initiate counter measure dispensing" function, while Figure 2.11 shows the operational implementation.

Planning Implementation			
Early	**Intermediate**	**Established**	**Advanced**
a. Guidance/plans document dispensing strategies (according to a tiered priority or alternate modality) to include: 1) open (public) PODs, 2) Closed PODs, and 3) Populations with Access and Function Needs.			
Written plans include none of the above	Written plans include one of the above	Written plans include two of the above	Written plans include all of the above
○	○	○	○
Comments:			

FIGURE 2.10. The planning implementation element of the "Identify and initiate counter measure dispensing" function.

Operational Implementation			
Early	**Intermediate**	**Established**	**Advanced**
a. Tiered priority or alternate dispensing modalities (as referenced in the planning element) have been exercised within the last five years.			
No exercise conducted	Tabletop exercise conducted	Functional exercise conducted	Full-scale exercise conducted or real incident
○	○	○	○
Comments:			

FIGURE 2.11. The operational implementation element of the "Identify and initiate counter measure dispensing" function.

As was previously described, both figures show the continuum of preparedness. The planning implementation has three items that should be achieved, and how many were actually achieved makes up the continuum. It is worth noting that in this case, all items are treated as equally important to the success of that function. The operational implementation, on the other hand, has only one item and the degree to which a plan has been tested makes up its continuum. According to the previously mentioned RAND report [40], most areas would only be able to claim, at best, an established level degree of readiness.

2.4. Computational Tools for Emergency Management

Attempting to validate response plans only through the use of live drills and training exercises is both prohibitively expensive in examining multiple different plans and may not examine the PODs at full capacity [40]. Thus, it has been necessary to use modeling and simulation to evaluate various aspects of emergency response. In fact, the Integrated Emergency Response Framework (iERF) developed by Jain et. al. sought to bring a holistic approach to modeling emergency responses. In their work, they described the need for a set of tools and standards that address a variety of emergency response needs. They stressed the importance of interoperability between currently available tools as well as support and training for responders and planners who will be using these tools [32, 31].

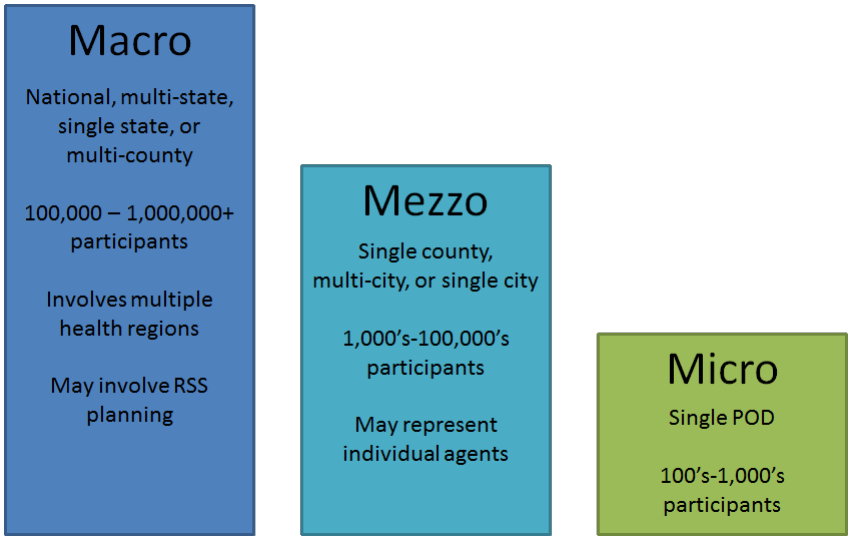


FIGURE 2.12. The macro, mezzo, and micro scales of analysis.

In addition to interoperability, the ability to simulate and examine plans from a variety of geographic scopes can provide emergency planners with a holistic view of a planned emergency response. From macro-scale simulations of national, state, or multi-county responses, to the mezzo-scale simulation of a single county's response plan execution, to the micro-scale level of the inner workings of a single POD, each scope sheds light on different facets of plans that can be useful for evaluation. Figure 2.12, provides some clarification by what is meant by macro, mezzo, and micro-scale modeling. At the macro-scale are simulations which examine emergencies that encompass hundreds of thousands to millions of people, and may involve one or more states or counties. At this scale, the geographic elements become increasingly important, such as ensuring that medicines are distributed efficiently to affected counties from the Strategic National Stockpile (SNS) to regional supply centers, and then to individual counties. This takes into account individual PODs and how and when supplies are delivered to them. For example, the RE-PLAN system, as developed by researchers in the Center for Computational Epidemiology (CeCERA) at the University of North Texas, can be configured to create and evaluate biological emergency response plans at this scale. It allows users to design plans for one or more counties at a time, which potentially allows for cross county resource sharing. This takes into account the populations in those areas, their distribution, and the road networks in the areas. Once a plan has been developed, RE-PLAN allows users to evaluate its feasibility and efficacy. This is performed by first determining if all of the PODs will be able to provide service for their respective populations, given a set of throughputs for each POD. After that, through the use of mathematical models, a user can analyze if the surrounding road networks will be able to adequately accommodate the traffic created by the activation of a plan. While this does provide a substantial level of analysis, it ignores dynamic problems during plan execution such as traffic accidents or POD closures. Further, the mathematical rates also ignore fine grain traffic dynamics that appear when using agent based models, such as the fact that vehicles may not travel at a constant speed at all time or the occurrence of gridlock [51].

Another example of planning and evaluation at the macro-scale is the RealOpt-

Regional system, developed by the Center for Operations Research in Medicine and HealthCare at the Georgia Institute of Technology. This online extension of the RealOpt framework allows users to design emergency response plans for large-scale regional needs. Similar to RE-PLAN, it examines population densities and demographics to better identify optimal POD locations. It can also examine traffic routes across a region to determine the best way to transport sick individuals, who have been selected by health officials at a POD, to be transported to hospitals. Once one or more plans have been generated through the system, RealOpt can also perform an economic model analysis of these plans to find a cost-effective response strategy. Further, RealOpt can model and simulate the logistics of Receipt, Stage, and Storage (RSS) facilities and Regional Distribution Nodes (RDN) for medical countermeasures, using RealOpt-RSS. This allows users to examine at a macro-scale how medications can be routed from the SNS to RSS facilities, and then further to individual PODs to achieve a rapid response and turn-around time [35, 36].

At the mezzo-scale, both RE-PLAN and RealOpt provide the same functionality as they do at the macro-scale, by examining a single county instead of an entire region. In addition to those systems developed via research in academia, several businesses have been created to address the need of computational tools for emergency management. The Advanced Disaster Management Simulator (ADMS), developed by Environmental Tectonics Corporation, was developed in the mid 90s in response to a need for training in airport fire rescue. This modular system has since been expanded to include training for wild fires, riots, industrial disasters, and biological emergencies to name a few. Primarily, ADMS trains managers and leaders to manage responders in the field during emergencies through scripted, dynamic simulations. One of their flagship products, ADMS-COMMAND, provides immersive, dynamic, and realistic training for incident command, in which emergency coordinators are organizing and deploying resources to the site of an emergency. The simulation itself can simulate large scale events and can take into account dynamic features of an emergency such as traffic and damage to facilities and roads. While this system does not specifically validate emergency response plans, it can be an effective way to examine popular strategies

in a variety of different scenarios from the perspective of being on the ground during the emergency [20].

Finally, examining emergency response at the micro-level frequently provides a highly detail oriented view of an emergency within a small area such as a portion of a city, a section of an area affected by the emergency, or inside a POD. For instance, DrillSim, as developed by Massaguer et. al., was constructed specifically test a variety of IT solutions during an emergency situation. It evaluates a particular IT configuration by modeling agents, the affected area, the crisis itself, and the infrastructure involved. It further provides a number of visualizations for users to examine, in both a 2-D and 3-D live renderings, how the drills are executing the planned response [6].

RealOpt also provides some POD evaluation functionality with RealOpt-POD. This system allows users to design and compare POD floor plans inter terms of usage and efficiency. It also analyses personnel usage and placement to determine how to minimize the average waiting time for individuals seeking treatment. Further, RealOpt-POD allows for simulations of the inner workings of a POD during POD activation, to allow for alternative analysis of multiple layout and staffing configurations [35].

There are also a variety of training simulators designed to provide emergency responders practice at working with the public during an emergency. The POD Game, developed by the Center for the Advancement of Distance Education at the University of Illinois at Chicago in conjunction with the CDC, is a simulation an biological emergency response to the intentional release of anthrax over a city. This simulator allows users to take on the role of a medical screener, forms reviewer, or medical countermeasures dispenser at an individual POD within the city. Your goal is to work through each videotaped individual seeking treatment at the POD and choose the correct options, while managing resources at your station and attempting to achieve the highest throughput rate possible [27].

2.5. Simulation and Modeling

Simulation and modeling (S&M) are widely used strategies "...to investigate, understand, or provide experiential stimulus to either (1) conceptual systems that do not exist or

(2) real life systems which cannot accept experimentation or observation because of resource, range, security, or safety limitations." We define a model as "[a] physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, or process" and a simulation as "[a] method for implementing a model over time...[and] a technique for testing, analysis, or training in which real-world systems are used, or where real-world and conceptual systems are reproduced by a model." [44] The distinction can be further made between models and simulations which are deterministic, those that do not include randomness in their processes, and stochastic, those that include randomness in their processes.

In the field of computation epidemiology, both stochastic and deterministic simulations and models play a crucial role in allowing for a variety of experiments to be performed on different populations. One of the most basic models is the Susceptible, Infectious, Removed compartmental model. In this model individuals move between the stages of a disease according the characteristics of the disease and population being modeled such as the disease's infectious period and the number of individuals a person comes into contact with at regular intervals. To simulate a disease outbreak using this model, differential equations can be used to examine how the outbreak will progress over time in the population. In [57], Shulgin, Stone, and Agar used a differential equation version of SIR to simulate the effect of using a pulse vaccination strategy over time in an effort to eradicate measles from a population. Stochasticity can also be added into a simulation by using Monte Carlo methods to randomly select starting points for the disease. This can be seen in [17] where experiments were conducted with varying vaccination distribution strategies across geographic regions using graph centralities. Multiple simulations were executed with different regions being initially infected to compare the different strategies.

Further, these type of models can be used to study vector-borne diseases, i.e. those carried by mosquitoes, ticks, or fleas. Because the population size of the vector can have a direct impact on the likelihood of a disease, such as Dengue, Malaria, or West Nile Virus, more accurate models of population size may lead to more accurate disease spread simulations. With regards to mosquito populations, temperature and rainfall have a direct impact on

mosquito breeding patterns. In [63], a mathematical model using differential equations was developed to take into account daily rainfall and temperature data on the breeding cycle of the *Aedes aegypti* mosquito. Once constructed and tuned, real data can be processed by the model to simulate how the populations will change per day over potentially many months.

2.6. Toward Automated Emergency Response Plan Validation

Based on this review, there is a clear need for the use of computational tools in the field of emergency management. In the case of RE-PLAN and RealOPT, these systems allow emergency manager and planner develop response plans more quickly and easily. However, while they do provide some evaluative processes, this can be expanded upon in terms of geographic scale and the addition of more realistic processes. Thus, this dissertation will present a new system that expands upon current emergency response plan evaluation and validation methodologies, while introducing new metrics and processes.

CHAPTER 3

VERPETS SYSTEM AND METRICS

In order to provide emergency managers and planners with a straight-forward and automated method to validate plan feasibility and efficacy this work presents Validation of Emergency Response Plan Execution Through Simulation, or VERPETS. This chapter describes the overall framework and its features, as well as how some of the challenges presented by different data elements were overcome. Additionally, metrics for evaluating plan feasibility and efficacy are presented.

3.1. Measures of Plan Feasibility and Efficacy

3.1.1. Response Plan Feasibility

While there exist multiple methods to determine whether a response plan is feasible or not, this dissertation will focus on one in particular: examination of how much time passed during a simulated response plan activation before all agents received treatment, i.e what was the completion time of the simulation. The inequality in Equation 3.1 provides the metric to determine whether a response plan is feasible. In this equation, the *Time Window* is a government mandated time limit by which all affected citizens must be vaccinated, *Population* represents total number of potentially affected individuals, *Processing Time* is the experimentally determined rate at which individual agents can receive medical countermeasures, B is a list of the number of booths for each POD, and B_i is the count of the number of booths at POD i. Thus, if the *Time Window* is greater than or equal to the calculated value, then the plan is deemed feasible.

$$Time\ Window \geq \frac{Population\ *\ Processing\ Time}{\sum_{i=0}^{|B|} B_i} \tag{3.1}$$

Further, while it may be sufficient to determine whether a plan is feasible or not, simply exceeding the time window by even a minute would technically disqualify a plan from

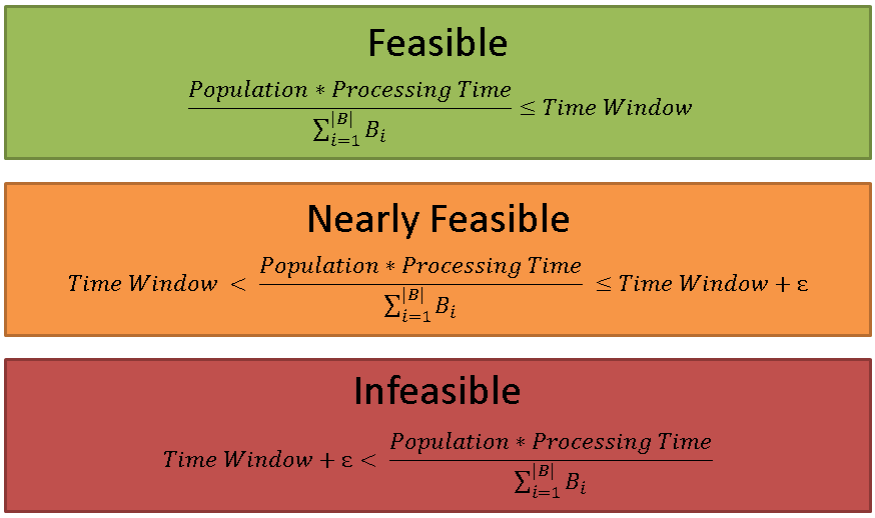


FIGURE 3.1. Three categories of response plan feasibility.

use. This could lead to the exclusion of potentially feasible plans due to random factors that could be mitigated if properly planned for, such as traffic accidents at critical road locations. Thus, an acceptable extension, ϵ, to the time window can be defined to prevent plans from being unnecessarily discarded. Figure 3.1 shows how this can be accomplished by utilizing a ternary, rather than a binary, categorization. As previously described, the *Feasible* category denotes a plan that completed below the *Time Window*, and the *Infeasible* category denotes a plan that completed above the *Time Window*. Additionally, there is a third category, *Nearly Feasible*, such that as long as the plan completed after the *Time Window* but before the *Extension*, it might be salvageable with proper mitigation efforts. This *Extension* would need to be determined by experienced emergency managers and planner and set to a duration they feel is appropriate. Adding in this extension also alters the *Infeasible* category by delaying its starting time boundaries until after the *Time Window* and *Extension*.

3.1.2. POD Throughput and Utilization

One of the key factors in determining if a plan will be feasible or not, is the throughput of each POD. POD throughput is a function of both the processing capabilities of the POD itself, and the utilization of the POD during any given time period throughout plan activation. A POD's processing capabilities are based on the number of booths a POD has available to treat agents and the processing time of a single patient at the POD. The more

active booths a POD has, the greater the number of individuals it can service concurrently, which leads to higher throughput. A POD's maximum throughput can be calculated via Equation 3.2, where *MPT* is the calculated maximum POD throughput for any POD, *Number of Booths* is the total number of booths a POD contains, and 1 over *Processing Time* is the rate at which a single person can be processed by the POD.

$$MPT = Number\ of\ Booths * \frac{1}{Processing\ Time} \tag{3.2}$$

However, this ideal maximum throughput cannot be achieved throughout the duration of plan activation. Individuals must first travel from their homes to the POD before they can be processed. This delay due to travel time causes changes in POD utilization over time. In order to calculate a POD's utilization at some timestep i during plan activation, first the POD's throughput at timestep i must be calculated using Equation 3.3. Here PT_i is the POD's throughput at timestep i, $Individuals_i$ is the number of people currently being processed at booths at timestep i, and *Processing Time* is the rate at which a single person can be processed by the POD. Finally, the POD's utilization at timestep i, PU_i, can be calculated by dividing the a POD's throughput at timestep i, by its maximum utilization, as seen in Equation 3.4.

$$PT_i = Individuals_i * \frac{1}{Processing\ Time} \tag{3.3}$$

$$PU_i = \frac{PT_i}{MPT} \tag{3.4}$$

As previously stated, a POD's utilization will change throughout the duration of plan activation. When a simulated plan activation begins, a POD's utilization will be 0, because all individuals begin at their homes. As individuals travel along the road network and arrive at the POD, utilization will increase. However, depending on the arrival rate, one of three

different scenarios will occur:

(1) the arrival rate will be on average greater than processing rate, and the POD's throughput will increase until it reaches the maximum
(2) the arrival rate will be on average equal to the processing rate, and the POD's throughput will increase until it reaches the maximum
(3) the arrival rate will be on average lower than the processing rate, and the POD's throughput will increase though will never reach maximum

Scenario 1 and 2 represent a POD that will, on average, achieve and maintain maximum utilization, while scenario 3 represents a POD that will have chronic underutilization. Achieving maximum utilization is preferable as long as the POD can process its population within the time limit. Doing so represents an efficient allocation of booths across PODs. Chronic underutilization in a POD may indicate that resources could be shifted to another POD in order to increase the other's throughput and process all of its individuals before the time limit.

3.1.3. Network Representation

In order to analyze POD utilization and perform experiments on the roads within a plan area, the road network along which cars travel to and from the POD must be examined. To do so, first the road network must be defined in graph terms. A plan can be represented by the graph G(V,E), with the vertices, V, being represented by the set of sets P,U,I and the edges, E, being represented by the set of sets S. In this case, P is the set of nodes representing PODs, U is the set of nodes representing geospatial units (GSUs), S is the set of edges representing each road segment, and I is the set of nodes representing the intersection of two or more road segments. In addition, weights, W, will be assigned to each road segment and can have different units of measure depending on the experimental setting. Some examples are road segment length, time to travel at an average speed, road segment capacity, or combinations of other factors. Table 3.1 shows an organized summary of the graph representation of a response plan. Further, a cut will be defined as the removal of a

road segment from G.

Variable	Definition
G(V,E)	The graph representation of the response plan
V	Set of P,U,I
E	Set of S
P	Set of all PODs
P_k	A single POD
U	Set of all GSUs
U_i	A single GSU
S	Set of all road segments
S_j	A single road segment
I	Set of all road segment intersections
W_{Sj}	Associated weight for a road segment
T_U	Total population of a single GSU, U
R_U	A subset of S that a T_U travels along from U_i to P_k and back.

TABLE 3.1. Definition of variables for network robustness.

One of the first methods of evaluating the road network is to examine the routes taken by individuals to and from their assigned POD. To begin, an optimal route from each GSU, U_i, to it's assigned POD, P_k, can be calculated using Dijkstra's algorithm [16]. This will take into account not just all of the road segments, S, but also the weight of each road segment, W_{Sj}. W_{Sj} in this case is the free-flow travel time, or how long would it take a care to traverse an edge given it's length and speed limit.

Once a route for each GSU has been generated, the total number of cars that would drive across a road segment during plan activation can be calculated using Equation 3.5. In this Equation, the total number of cars for a GSU was added, T_U, to a single road segment, S_j, if that road segment is actually part of that GSU's route, R_U. Summing this conditionally over all GSU's, provides the total number of cars.

$$\textit{Total cars on } S_j = \sum_U T_U[S_j \cap R_U \neq \emptyset] \tag{3.5}$$

Once this total is known for each road segment, multiple analyses can be done. When joined with a shapefile that contains matching road segments, maps of the traffic flow can

be generated to better visually understand where critical portions of the network might be. Figure 3.2 is an example of this type of analysis. Segments of the road network that transport low numbers of people over the course of the plan activation darker green, which those that transfer high numbers of people are darker red. High levels of traffic flow along the network is typically seen around the POD, represented by the red triangle, as cars converge on the road leading to the POD.

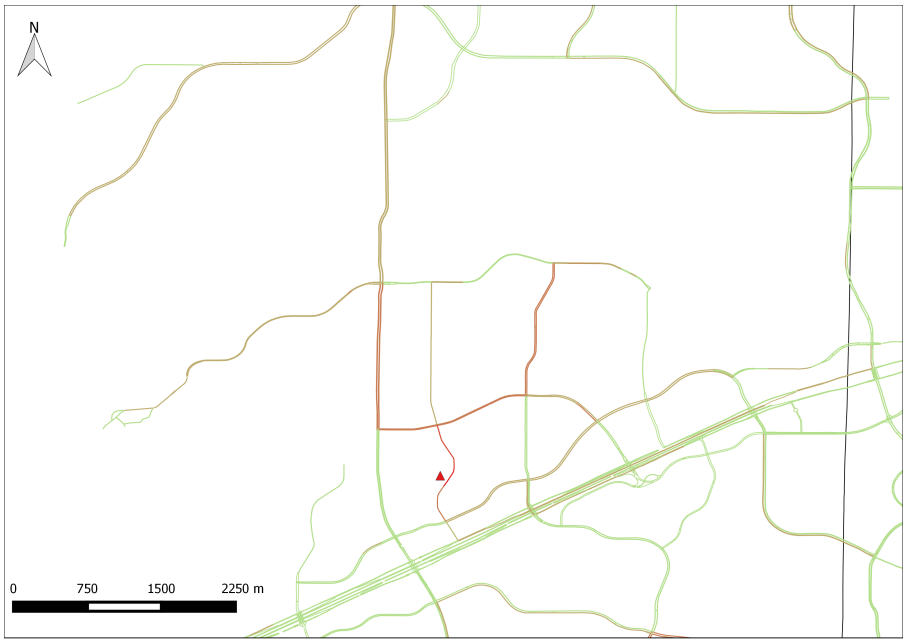


FIGURE 3.2. A road network analysis for a single POD in Denton County.

3.1.4. Traffic Flow and Congestion

Originally, this research had sought to show that removing road segments from the network could cause congestion that would ultimately delay plan completion past the time limit. However, there were two key problems with this approach:

(1) Selecting more than one road segments to remove from the network leads to a combinatorics nightmare

(2) Removing small numbers of road segments has either little to no effect, or the plan is an instant failure due to a POD becoming unreachable by members of its catchment area.

For example, one of the counties used for simulation, Rockwall County, TX, has 2,419 road segments after its road network has been processed. That results in 2,419 choices for the removal of a single road segment, 2,924,571 choices if two segments are removed, 2,356,229,369 choices if three segments are removed, etc. This situation is untenable when each simulation takes roughly three minutes to complete. Additionally, Rockwall County is the smallest county selected for simulation. Denton County, TX has 21,517 road segments after its road network is processed, and each simulation takes roughly 7 minutes to complete. Due to this combinatorial challenge, the original problem was reevaluated.

Upon further investigation of preliminary results, it was determined that the reason for this binary outcome when removing small numbers of road segment was caused by the network itself. Due to the U.S. road network's robustness and availability of easy to reach, alternative routes, vehicles can usually find a new route with a minimal to no delay in the event of a non-disconnecting segment being removed. However, in the event that removal of a road segment disconnects a POD from its catchment area, the plan immediately becomes *Infeasible* as some portion of the population would never be able to reach or leave the POD. This was the case for PODs that were placed along one-way, highway feeder roads. Removal of any segment of the feeder road, either before or after the POD, would prevent flow in and out of the location. Figure 3.3 shows one such instance in a plan for Rockwall County. Removal of any of the three road segments would cause plan failure. While this information could be helpful to emergency managers and planners, it would not be surprising to suggest that PODs on one-way roads need extra precautionary measures.

In addition to studying how the removal of small numbers of road segments affects plan outcomes, another dynamic affecting POD throughput was examined. As previously mentioned, the relationship between the arrival rate and the processing rate of a POD affects POD utilization. In the event that the average arrival rate is greater than the processing rate, cars will begin to queue at the POD. This is significant because if the arrival rate is maintained throughout the simulation, once cars beginning queuing the queue will not become empty until the last waiting car is processed. We call this situation of a perpetually

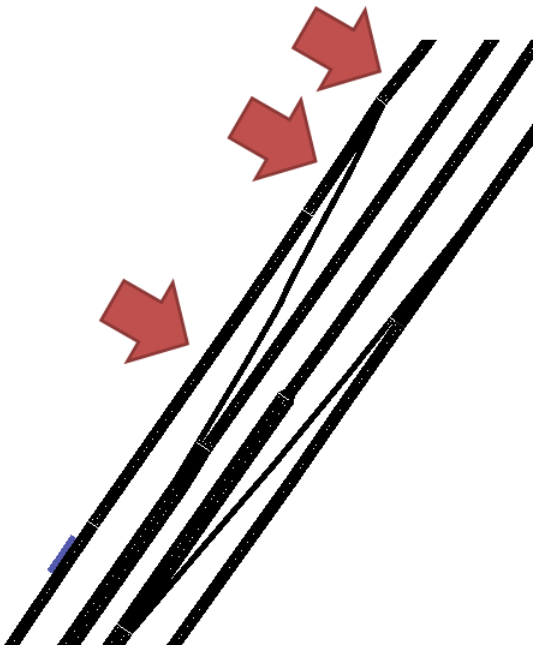


FIGURE 3.3. Road segments whose removal would cause disconnection of a POD and portions of its catchment area.

expanding queue at the POD runaway congestion. The impact of runaway congestion is a significant decrease in the relevance of travel-time on plan *Feasibility*, as the POD processing rate becomes the rate determining step due to the POD being at perpetual maximum utilization.

Further, if the POD's average arrival rate is equal to the processing rate, then the POD may experience controlled congestion. Controlled congestion can either be in the form of a queue that maintains a constant size, or one that oscillates between being empty and being filled to some maximum size. In both cases, the POD is still at perpetual maximum utilization, and thus again, the importance of travel time on the plan's outcome is diminished.

Equation 3.6 describes the limit of a booth's ability to avoid runaway congestion. If the sum of the number of cars released from all GSU's in a catchment area is less than the rate of processing for the POD, then runaway and controlled congestion are avoided. If the sum is equal to the processing rate, then either controlled congestion or no congestion will occur. Finally, if the sum is greater than the processing rate, the runaway congestion will occur, though the size of the waiting queue will depend on how much faster cars are arriving

than the rate of processing.

$$\frac{Number\ of\ booths}{Processing\ Time} \geq \sum_i \frac{Number\ of\ cars\ released\ from\ U_i}{Processing\ Time} \tag{3.6}$$

While Equation 3.6 does provide one limit, it lacks the effect of travel time, which plays a key role in the ability for a POD to maintain maximum utilization. For example, if a POD can process 10 cars in 3 minutes, and a total of 10 cars are being released from GSUs at different distances from the POD, then the POD may experience under utilization at the beginning, but congestion later on as the further away cars finally arrive. Thus, Equation 3.6 is reformulated into Equation 3.7 to account for this delay. This Equation recognizes different GSU's account for different time costs in terms of load on the POD. If again, there was a POD that can process 10 cars in 3 minutes, and it takes 3 minutes for a car to reach the POD from its GSU, then 20 cars could be released at once while still avoiding runaway congestion.

$$\frac{Number\ of\ booths}{Processing\ Time} \geq \sum_i \frac{Number\ of\ cars\ released\ from\ GSU_i}{Processing\ Time\ +\ Travel\ Time\ from\ GSU_i\ to\ POD} \tag{3.7}$$

Therefore, Equation 3.7 now provides a hard upper-bound to the release rate of vehicles from GSU's to their POD, that prevents runaway congestion. However, the Equation still does not capture realistic traffic dynamics that occur when traveling on the roadway. It ignores the need for acceleration, deceleration, and traffic laws that coerce drivers to yield to other motorists at intersections. Thus, in order to better capture these details and analyze their effects, a simulation of these driving patterns is necessary in order to perform an accurate analyses.

3.1.5. Schedulability

Due to how plan feasibility is affected by POD throughput and the POD arrival rate, a relationship can be formalized. Specifically, because each agent must arrive at a POD

and be processed, and because the POD has a finite number of booths there must be some order that the population of a catchment area must be processed in. Thus, a schedule for processing can be formed based on that ordering. In this case, it can be said that a plan is schedulable if an ordering can be placed on the population of a catchment area such that the population is processed within the time limit. Therefore, there exists a relationship between the feasibility of a plan and the schedulability of a plan, which is presented in Theorem 1 with its associated Lemmas. Table 3.2 provides a list of symbols and their meanings used in Theorem 1, Lemma 1, and Lemma 2.

Variable	Definition
F	A feasible emergency response plan.
C	Set of catchment areas for an emergency response plan.
c	A single catchment area in C.
P	The set of all PODs for an emergency response plan.
p	A single POD in P.
m	The number of booths at a single POD p.
A	The population being treated by an emergency response plan.
a	The population of a single catchment area c.
a_o	A single agent i in catchment area c.
H	The set of schedules for all PODs in an emergency response plan.
h	A schedule for a single POD p.
h_t	The index, t, of a single time slot in an schedule h.
θ	The time limit for an emergency response plan.
ϕ	The average processing time for a single individual at a POD.

TABLE 3.2. Definition of variables for network robustness.

THEOREM 1. An emergency response plan is schedulable if and only if it is feasible.

LEMMA 1. If an emergency response plan is schedulable, then it is feasible.

PROOF. Let h be an empty schedule for the servicing of population, a, of a catchment area, c. Let h_t be the index to any time slot in h, such that each slot represents a single, schedulable time unit and $|h| = \theta$. Additionally, place an upper-bound on h_t such that the value at that index cannot be greater than the m at p, the POD servicing that catchment area. An agent a_o is scheduled in h to be processed from time slots h_t to h_u, such that $h_t + \phi = h_u$, only

when $h_u \leq \theta$ and no slot between h_t and h_u has more agents scheduled than m.

For example, assume $\phi = 5$ seconds, $\theta = 14$ seconds, and $m = 3$. Figure 3.4 represents h, an empty schedule. Attempting to schedule an agent results in one of three possible scenarios.

(1) Sufficient space exists from h_1 to h_ϕ and so the agent can be added there at the beginning of the schedule. To schedule an agent, ϕ number of slots in the schedule are incremented by 1, as shown in Figure 3.5.

(2) If any of the slots from h_1 to h_ϕ equal m, then the schedule must be searched for a gap ϕ slots long. In the case of Figure 3.6, three agents have already been scheduled, and thus a forth agent must find a gap further in the schedule.

(3) After a certain number of agents are added, the schedule becomes full, as seen in Figure 3.7, and no gaps of size ϕ or larger remain. Attempting to add an agent will fail, as that would cause the agent's completion time to exceed θ and cause the plan to fail.

0	0	0	0	0	0	0	0	0	0	0	0	0	0
h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	h_{12}	h_{13}	h_{14}

FIGURE 3.4. An empty schedule.

1	1	1	1	1	0	0	0	0	0	0	0	0	0
h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	h_{12}	h_{13}	h_{14}

FIGURE 3.5. A schedule with a single, scheduled agent.

3	3	3	3	3	1	1	1	1	1	0	0	0	0
h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	h_{12}	h_{13}	h_{14}

FIGURE 3.6. A schedule with four, scheduled agents.

Therefore, if all a_o in a can be scheduled in h, then each a_o successfully received treatment before θ within c. If all c in C has a schedule, such that for all a_o in a successfully

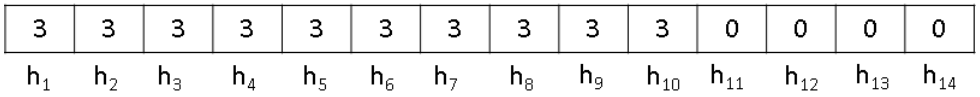


FIGURE 3.7. A schedule with the maximum allowable number of scheduled agents.

received treatment before the θ, then the emergency response plan is feasible. Therefore if a plan is schedulable, then it is feasible. □

LEMMA 2. If an emergency response plan is feasible, then it is schedulable.

PROOF. Let there be a plan that such that for each c, Equation 3.8 holds. If this relationship holds, then each agent in each catchment area must be able to receive treatment at a POD for the duration of the processing time. This also means that no more than booth number of agents can be at a POD receiving treatment concurrently due to the physical limitation of the POD. Therefore, there must exist some ordering of agents being processed for each POD that allows all agents to be processed before the time limit. A distinct ordering can be interpreted as a scheduling. Therefore, if a plan is feasible, then it is schedulable. □

$$\frac{\phi}{m} * a + f(travel) \leq \theta \tag{3.8}$$

PROOF. Lemma 1 proves that if a plan is schedulable then it is feasible, and Lemma 2 proves that if a plan is feasible then it is schedulable. Thus, it can be said that a plan is schedulable if and only if it is feasible. □

3.1.6. Release Rates

In theory, it is possible to compute a schedule h for a feasible response plan such that the waiting time for each agent is zero and the booths operate at maximum utilization. If all agents for a POD are released at the same time, some agents will necessarily have to enter a queue to wait for service. Due to the delay caused by the processing time, the arrival rate will overcome the processing rate, thus leading to congestion at the POD. Further, this queue will not be empty until there are no more agents awaiting service. This congestion

Variable	Definition
P	The set of all PODs for an emergency response plan.
p	A single POD in P.
n	The number of census block groups assigned to POD p.
B	A set of all census block groups assigned to p.
b_j	A single census block group assigned to p where $j \in \{1, 2, ...n\}$.
A	The population being treated by an emergency response plan.
a	The population of a single catchment area c.
a_o	A single agent o in catchment area c.
m	The number of booths at POD p.
l_i	A single booth at p where $i \in \{1, 2, ...m\}$.
t_j	Travel time to reach POD p from census block group b_j
b_{ik}	The k_{th} census block group which has been assigned booth l_i
t_{ik}	Travel time to reach POD p from census block group b_{ik}
s_i	Number of block groups which have been assigned booth l_i
θ	The time limit for an emergency response plan.
ϕ	The processing time for a single individual at a POD.

TABLE 3.3. Definition of variables for optimal release rates.

can be avoided by controlling the release rate of the agents at each census block group.

For simplicity, the release rates are computed for the census block groups corresponding to a single POD p. The process is independent of the number of PODs and therefore is generalizable for the entire plan. Additionally, assume that the number of agents at each census block group is equal, thus, the census block groups differ only in their travel time to the POD. The non-uniform population scenario is an extension to this methodology and is discussed later. Table 3.3 provides a list of parameters and their notations used to compute the release rates. For this calculation, two cases need to be considered. Case 1 represents the typical case in which the number of census block groups, n, is greater than or equal to the number of booths m. Case 2 represents the case in which the number of census block groups, n, is less than the number of booths m. It will be shown that Case 1 can be later dissolved into Case 2 as the population for census block groups are processed.

Case 1. $n \geq m$

The agents from each census block group are assigned a booth at the corresponding POD, such that all agents from a census block group go to the same booth and the booths are

assigned to the agents from census block groups based on similarity (closeness) in travel time. Formally, the assignment of agents to booths has to satisfy the following constraints:

(1) All agents from a census block group are assigned the same booth at p

(2) If agents from census block group b_j and agents from census block group b_k are assigned the same booth l_i this implies that $|t_j - t_k| = min\{|t_j - t_l|\}\ \forall\ l \in \{1, 2....n\}$. In other words, n census block groups are divided into m clusters and t_j is used to group similar census block groups.

Let X_i be the set of census block groups assigned to a booth l_i at p such that this assignment satisfies the above two conditions and the block groups are ordered in ascending order by travel time within this assignment. Thus, $X_i \subset B$ and $X_i \neq \emptyset$. Further, let the elements of X_i be defined as b_{ik}, where $b_{ik} = b_j$ if b_j is assigned booth l_i and $t_{i1} < t_{i2} \ldots t_{ik} \ldots t_{is_i}$ for $k \in \{1, 2...s_i\}$ where $s_i = |X_i|$. Each booth l_i is assigned agents one at a time from $b_{i1}, b_{i2} \ldots b_{is_i}, b_{i1}$ so on. In order to ensure that the plan executes without congestion, an agent should arrive at the booth when the previous agent is about to leave i.e. the arrival time for the next agent should be equal to departure time of the current agent.

For booth l_i, the first agent leaves from block group b_{i1} at $t = 0$, reaches the booth l_i at $t = t_{i1}$, and leaves the booth at $t = t_{i1} + \phi$. To achieve maximum utilization of a booth, the next agent (from b_{i2}) has to arrive at $t = t_{i1} + \phi$ and hence leave its block group at $t = max\{t_{i1} + \phi - t_{i2}, 0\}$. For an agent a_o, let the time of departure from its block group be denoted by $t_{db}(a_o)$, and the time of arrival at its POD be $t_{ap}(a_o)$ and the time of departure from the POD be $t_{dp}(a_o)$

Table 3.4 shows the arrival and departure time for subsequent agents arriving at booth l_i. The release rate for the agents from b_{ik} can be calculated as the difference between the time of departure of an agent from b_{ik} and the the time of departure of the previous agent from b_{ik}. From Table 3.4 we observe that $t_{db}(2s_i + 1) - t_{db}(s_i + 1) = s_i\phi$ which is the release rate for b_{i1}. Similarly, the release rate for $b_{i2} = t_{db}(2s_i + 2) - t_{db}(s_i + 2) = s_i\phi$. Thus, the release rates for all the census block groups which have been assigned l_i is $\frac{1}{(s_i\phi)}$, where s_i is the number of block groups which have been assigned booth l_i and ϕ is the processing

Agent	Block	$t_{db}(a_o)$	$t_{ap}(a_o)$	$t_{dp}(a_o)$
1	b_{i1}	0	t_{i1}	$t_{i1}+\phi$
2	b_{i2}	$max\{t_{i1}+\phi-t_{i2},0\}$	$t_{db}(2)+t_{i2}$	$t_{db}(2)+t_{i2}+\phi$
3	b_{i3}	$max\{t_{dp}(2)-t_{i3},0\}$	$t_{db}(3)+t_{i3}$	$t_{db}(3)+t_{i3}+\phi$
.	.	.	.	.
.	.	.	.	.
s_i	b_{is_i}	$max\{t_{dp}(s_i-1)-t_{is_i},0\}$	$t_{db}(s_i)+t_{is_i}$	$t_{db}(s_i)+t_{is_i}+\phi$
s_i+1	b_{i1}	$t_{dp}(s_i)-t_{i1}$	$t_{dp}(s_i)$	$t_{dp}(s_i)+\phi$
s_i+2	b_{i2}	$t_{dp}(s_i)+\phi-t_{i2}$	$t_{dp}(s_i)+\phi$	$t_{dp}(s_i)+2\phi$
.	.	.	.	.
.	.	.	.	.
$2s_i$	b_{is_i}	$t_{dp}(s_i)+(s_i-1)\phi-t_{is_i}$	$t_{dp}(s_i)+(s_i-1)\phi$	$t_{dp}(s_i)+(s_i)\phi$
$2s_i+1$	b_{i1}	$t_{dp}(s_i)+(s_i)\phi-t_{i1}$	$t_{dp}(s_i)+(s_i)\phi$	$t_{dp}(s_i)+(s_i+1)\phi$
$2s_i+1$	b_{i2}	$t_{dp}(s_i)+(s_i+1)\phi-t_{i2}$	$t_{dp}(s_i)+(s_i+1)\phi$	$t_{dp}(s_i)+(s_i+2)\phi$
.	.	.	.	.
.	.	.	.	.

TABLE 3.4. Time line for agents at booth l_i

time for each agent at the POD.

Case 2. $n < m$

In the scenario where the number of booths, m, at the POD is greater than the number of census block groups in the catchment area corresponding to the POD, the agents from the n blocks can be assigned to the first n booths and released at a rate of one agent per ϕ time units. To make use of the remaining $m-n$ booths the release rate of one of the blocks can be increased by $\frac{\phi}{m-n+1}$. It should be noted that the agents from this block groups will be processed faster than the other block groups. Therefore, the booth assigned to this block group will run out of agents to process more quickly, and thus can be used to process agents from another booth. Then, the release rate of another census block group with agents remaining to be processed can be increased by $\frac{\phi}{m-n+2}$ and so on.

The same methodology can be applied to non-uniform population census block groups, with the only difference being s_i for each l_i is dynamic. s_i will decrease by one, once all agents from a block group in X_i have been processed, thus increasing the release rate of the remaining census block groups. If all agents for a booth l_i have been processed then this booth can be used to increase the release rate of a block from another booth similar to case 2 above.

Alternate Release Strategy It is possible to further maximize utilization at each booth l_i by altering the sequence of agent releases. When using a round-robin approach, in the first cycle, it is possible that an agent from b_{ik} is processed before the agent from $b_{i(k+1)}$ arrives when $t_{i(k+1)} > t_{ik} + \phi$. This will cause the booth to be idle for $t_{i(k+1)} - (t_{ik} + \phi)$. By switching from a round-robin approach to a shortest travel time approach, i.e. from $b_{i1}, b_{i2} \ldots b_{is_i} b_{i1}$ to $b_{i1}, b_{i1}, b_{i1} \ldots b_{is_i}, b_{is_i}, b_{is_i}$, this will prevent any idle time at the booth after the initial travel time, t_{i1}, if agents are released from each block group at a rate of $\frac{1}{\phi}$. However, this causes all block groups in X_i to be processed sequentially instead of concurrently.

3.2. Simulation

One of the first major decisions before designing the VERPETS system was to select an appropriate traffic simulator with which to simulate the activation of emergency response plans. Rather than attempting to build a new simulator and thus having to construct traffic models, sensors, a data management system, and a visualization system, currently available simulators and simulation environments were evaluated. For these experiments, the necessary features were:

- Accessibility
- Compatibility
- Sustainability
- High Rates of Concurrent Agent Management
- Effective Visualization

3.2.1. Necessary Features

First and foremost, accessibility was important in determining a suitable simulation environment. What is meant by accessibility is that the simulation environment to be used for this system needed to be free and open-source. Since this system will eventually be included in the RE-PLAN suite, it is important that the simulation environment not add unnecessary costs to the use of RE-PLAN.

In addition to lower long term costs, being open-source provides a number of advan-

tages. According to the Open Source Initiative [52], to be considered open-source software, a released program must: allow free redistribution; provide the entire set of source code used to compile the software and make sure it is easily accessible; allow modification to the code and allow for derived works that would be distributed under the same license agreement of the original software; not discriminate against any person, any group, or any field of endeavor; allow the original license to be distributed with the software and/or any pieces of the softwares package and must not restrict additional software; and the license must not dictate the type of technology the software can be used with. By prioritizing open-source software with the previously listed features for the simulation environment, it can be assured that the software can be modified to fit RE-PLANs diverse needs while being able to be distributed to clients without worry about licensing issues.

Next, compatibility was key, due to the nature of how the simulation will be integrated into, and function with, RE-PLAN. Since RE-PLAN needs to be deployable on any computer, one of the initial design choices was to program the system in Java [51]. With that in mind, the simulation environment used for this research needed to be easily deployable with, and executable from within, RE-PLAN. A simulation environment that needed no prior compiling or was based on an interpreted language would be preferable.

Additionally, sustainability was essential for the long term use of the simulation environment with RE-PLAN. Since the environment is a third-party piece of software and not being actively maintained by any of the RE-PLAN team, it is vital that the environment be in a state of active development. This ensures that the simulator stays up-to-date with the latest computer security, operating system, and programming language updates. However, while it may necessitate periodic updates to the module that executed the simulator, the long term benefit of not having to contend with the simulators internal integrity is more valuable.

Another critical feature is the ability for the simulator to handle high rates of concurrent agents. Since RE-PLAN primarily designs plans at the county level, the simulator must be able to simulate executions at the same scale. This will require it to be able to

manage at least two million concurrent vehicles, in order to be useful for some of the largest counties in the country, such as Dallas county with 2.5 million residents, Orange county with 3.1 million residents and Harris county with 4.4 million residents [60]. It also needs to be able to do in such a way that the use of an embarrassingly parallel setup could be utilized to decrease runtime.

Finally, a clear and effective visualization of the simulation is necessary for response planners to better understand the reasons why plans are failing during a simulated activation. Identifying where unforeseen bottlenecks in the traffic flow might occur is key for their prevention or mitigation. Additionally, such findings can be more effectively relayed to decision and policy makers through a live demonstration as opposed to a basic description or purely numerical representation.

3.2.2. Potential Simulation Environments

With these features in mind, several potential simulation environments were examined. Some of the most popular environments that were discovered were Repast Simphony, TRANSIM, MATSim, and SUMO. Each of these options was examined with respect to all of the necessary features.

Repast Simphony is the modeling and simulation client included in the Repast Suite developed by Argonne National Laboratory. The suite boasts over 15 years of continuous development, and is free and open-source. However, while the models and simulations can be generated in Java, significant time would be required to develop the simulation for this research. Because one of the desired characteristics for a simulator in this research was that it be sustainable outside of the RE-PLAN team, Repast Simphony was deemed unsuitable [5].

The Transportation Analysis and Simulation System (TRANSIMS) is a transportation simulation toolset developed by the University of California and Los Alamos National Laboratory. Unlike Repast Simphony, TRANSIMS is specifically a transportation simulation system and has been used for emergency response plan evaluation [65]. It appears to achieve all of the necessary features to be used in this research with the exception that it

is difficult to determine if it is being still being maintained. While it appears that newer version are in development they do not appear to be available for public use, which leaves only the 2013 4.0.8 version available [28]. For this reason, despite being otherwise qualified, TRANSIMS was not used for this research.

Another potential simulation system, Multi-Agent Transportation Simulation (MATSim) is a joint effort towards transportation modeling between the Institute for Land and Sea Transport Systems at Technische Universität Berlin, the Institute for Transport Planning and Systems at the Swiss Federal Institute of Technology Zurich, and Senozon. MATSim met all of the necessary features that were previously described as well as being a widely used choice for vehicle simulations and routing problems [37, 30, 38]. The main limitation was the need to install third-party extensions to obtain the desired functionality for activating a response plan. MATSim should remain as a back-up solution in the case that SUMO fails to continue to meet these standards.

Finally, the simulation engine that was selected for this research was Simulation of Urban MObility (SUMO), courtesy of the Institute of Transportation Systems in Berlin. This engine boasts high portability, dynamic routing, and it is free and open-source. Additionally, it requires only a simple set of XML files to setup an experiment, and provides a concise XML formatted output that can be configured to display all or a selection of vehicles in the output. Further, it has been used to simulate the traffic patterns in the city of Cologne, Germany, containing roughly one million people, during a visit from the Pope and during the 2006 Soccer World Cup, so it can clearly function with a large number of concurrent agents [29, 34]. For these reasons, SUMO, specifically version 0.32.0, was selected as the simulation engine for this research.

3.3. Validating Emergency Response Plan Execution Through Simulation

As seen in Figure 3.8, the VERPETS framework is divided into three major components: the Simulation Manager, the Data Manager, and the Master Controller. In order to provide a more modular approach to providing functionality, each component is specialized in its responsibilities. The Master Controller component facilitates information transfer and

organizes tasks for the Simulation Manager and Data Manager. It also is responsible for the initialization of the VERPETS system via a configuration file, reporting final results, and the final cleanup of files created during program execution.

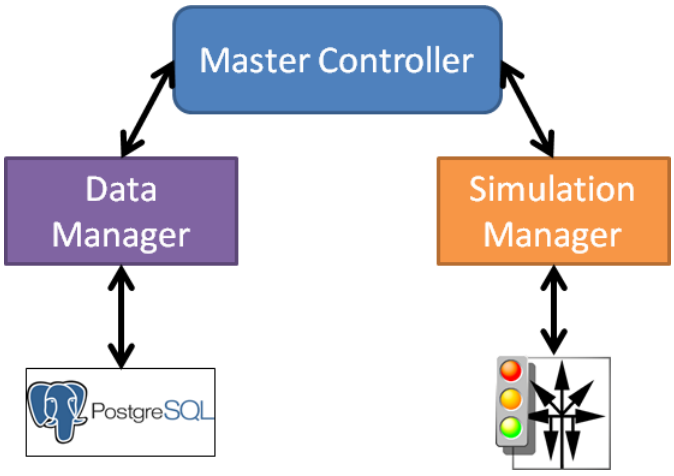


FIGURE 3.8. The three major components of the VERPETS system.

The Data Manager component functions as a centralized data storage and quick-access location for the VERPETS system. Configuring the data management of the framework this way has two main benefits. First, it reduces redundant data storage within the system during operation. Due to the large size of some of the emergency response plans and their associated road networks as well as the simulation data associated with them, repeated copies of this information could cause the program to exhaust available memory. Second, centralizing data storage improves data flow by providing a single access point to all of the information needed to create simulations and generate comparisons of different emergency response plans.

The Data Manager maintains three storage locations, local memory, disk, and a PostgreSQL database. The majority of the information used by the system to generate and evaluate experiments is stored in local memory. Some of the data must be processed by outside softwares, and thus is stored in XML formatted files. Finally, the database stores both GIS data and the emergency response plans developed by RE-PLAN. The use of the PostGIS extension to the database facilitates certain GIS related calculations, such as centroid determination, that will be described later.

Finally, the Simulation Manager component is the core of the system's ability to

evaluate emergency response plans. It handles the generation of the files needed by the simulator, the execution of one or more simulations, and the recording and processing of simulator output data. The Simulation Manager also performs the static flow analysis.

Figure 3.9 presents a flow diagram of the major tasks performed by VERPETS. The first task is to read a configuration file which initializes multiple variables in the system. The road network associated with the emergency response plan must then be processed, as will be discussed later. A biological emergency response plan (BERP) object must then be created using the road network and the emergency response plan generated via RE-PLAN. The static flow analysis, as described in subsection 3.1.4, must then be performed as the road network usage data will be used later. Next the emergency response plan will be simulated to determine if it is feasible, nearly feasible, or infeasible. Finally, the results of all analyses will be reported to the user. The remainder of this section will describe how each of these tasks are performed by the VERPETS system and its modules.

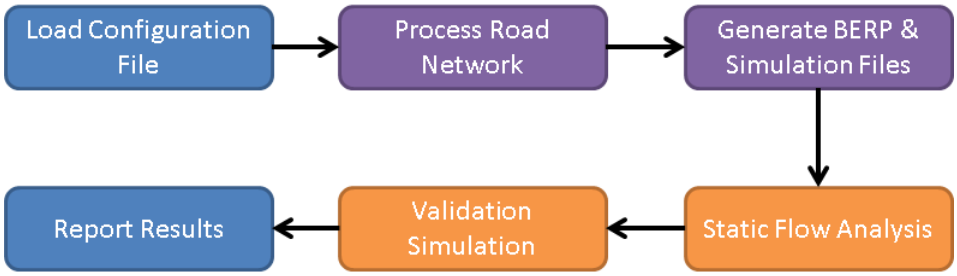


FIGURE 3.9. A flow diagram of the major tasks in VERPETS.

3.3.1. Generating Emergency Response Plans

To generate each plan, first, the desired county was selected from the RE-PLAN database and loaded. Next, a sufficient number of booths were added to the plan such that the desired estimated completion time was achieved. The number of PODs, and thus catchment areas, were determined by attempting to distribute an even number of booths, in this case 15, to each POD. Once this information was obtained, the catchment areas were generated via an equal population partition (EPP). PODs were then automatically assigned via RE-PLAN to available POD locations. Finally a closest POD partition (CPP)

was performed to minimize travel time and route overlaps. This plan was then saved to the database to be used by VERPETS.

3.3.2. Loading the Configuration File

One of the first tasks that need to be performed is the initial configuration of VERPETS. This task is the responsibility of the Master Controller, and involves reading the configuration file, creating many of the files that will be needed by the other modules and programs, and initializing many of the global variables. A list of the inputs required, along with a description of each, can be found in Table 3.5.

Input	Description
OSM File	The location of the .osm file associated with the plan to be simulated
Response Plan Name	The name of the response plan generated in RE-PLAN which is to be examined.
Plan Time Limit	The time limit, in hours, in which a plan must treat all of its population.
Average Processing Time	The time, in seconds, for how long it takes to process a single agent.
Vehicle Release Interval	The delay, in second, between vehicle releases onto the network.
Households Per Agent	The number of individuals represented by a single agent.
Rounding	Determines how to round the number of individuals represented by a single agent. Can be set to always ceiling, floor, round to the nearest whole number, or no rounding.
One Hour Cutoff	Should the simulation halt after an hour of simulated time and extrapolate the completion time.
Batch Size	The number of catchment areas that will be processed in parallel.

TABLE 3.5. A list of the configurable variables in the system, and an explanation of each.

The first four configurable variables concern details about response plan generation, such as the road network, the name of the plan, the time limit for the plan, and how long it takes to process a single agent. The remaining configurations apply to the simulation of emergency response plan activation. The vehicle release interval controls how often vehicles are queued for release from a census block group and can be set to any nonnegative integer.

Households per agent is a scaling mechanic that determines how many individuals a single agent represents in the simulation and can be set to any positive integer. This also affects the speed at which cars can travel along the road network and how long they take to be processed at the POD. The rounding configuration is only applicable when a greater than 1:1 scaling has been applied, and determines how to round when the number of individuals in a census block group do not evenly divide by the scaling factor. This can be set to 1,2,3 or 4, which correspond to a ceiling strategy, a floor strategy, a closest strategy, or no rounding respectively. The one hour cutoff can be used when using a 1:1 scaling and specifies whether or not VERPETS should stop the simulation after one hour of simulated time and mathematically estimate the completion time of the plan. This can be set to either true or false. Finally, the batch size is used improve performance by running many of the processes used in simulation file generation, as well as the simulations themselves, in parallel. This can be set to any positive integer, though care should be taken against setting the size too high or the computer system may be overwhelmed by the number of concurrent processes. Each of these configurations will be discussed in greater depth below.

3.3.3. Processing the Road Network

Once the configuration file is loaded, the Data Manager can begin processing the road network to be used in simulations. Figure 3.10 represents the data flow associated with this process. First, as described previously, a user will download an OSM file of the area being simulated. While the OSM file does provide the road network for the area, it cannot be used in its raw format. The simulation environment SUMO, provides a tool called NETCONVERT, that can take in an OSM file and convert it to a proprietary network file. This process breaks an OSM way into its individual road segments for easier simulation. However, in order to potentially aggregate the data from the segment level back to the way level, the Data Manager renames every way with a unique ID number, as that data is transfered to each of the road segments during processing. The actual execution of NETCONVERT is controlled by the Simulation Manager and coordinated by the Master Controller. Since many drivers rely on large road arteries when driving, this assumption was taken into account by

NETCONVERT, whereby only highway roads that were labeled by OSM as primary, secondary, tertiary, motorway, or trunk were retained for the actual simulation. Additionally, any segment that was not attached to the majority of the network was deemed isolated and removed.

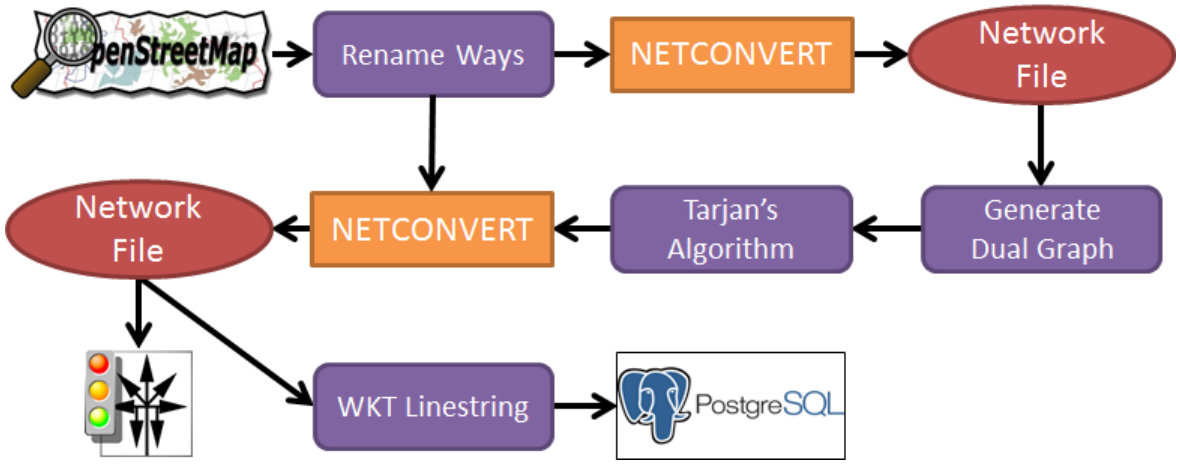


FIGURE 3.10. Work flow for processing OSM data.

However, while removing isolated edges prevented improper assignments of road segments later on, it did not account for the problem of one-way segments that do not connect back to the rest of the network. This was discovered early on to be an issue incurred by using only a portion of the road network for the simulation. While the majority of the U.S. road network has multiple routes to connect any road to other roads, by only selecting a portion of the road network and removing lower class roads, some of the highway class roads ended in a one-way segment that had no turn-around. Thus, if a driver were to be assigned that segment as the beginning point of their route, they could never reach their destination. In order to solve this problem, Tarjan's strongly connected components algorithm [58] was employed to remove any road segment that could not reach or be reached by every other segment. Additionally, Tarjan's was selected over Kosaraju-Sharir's algorithm [56, 3] because Tarjan's only requires a single depth first search traversal of the graph, at the cost of additional memory requirements, as opposed to Kosaraju-Sharir's required double depth first search traversal. Algorithm 1 shows Tarjan's algorithm.

Additionally, because Tarjan's algorithm requires a list of verticies, the road network

Algorithm 1: Tarjan's Strongly Connected Components

```
   Input  : Digraph G=(V,E)
   Output: List of strongly connected components
 1 Integer i = 0;
 2 Stack s = empty stack of verticies;
 3 List components = empty list of components;
 4 Function STRONGCONNECT(v):
 5 |   v.lowpoint = v.lowlink = v.number = i;
 6 |   i = i+1;
 7 |   s.push(v);
 8 |   for each w adjacent to v do
 9 |   |   if w.number == null then
10 |   |   |   STRONGCONNECT(w);
11 |   |   |   v.lowpoint = min(v.lowpoint,w.lowpoint);
12 |   |   |   v.lowlink = min(v.lowlink,w.lowlink);
13 |   |   else if w is ancestor of v then
14 |   |   |   v.lowpoint = min(v.lowpoint,w.number);
15 |   |   else if w.number < v.number AND w is in s then
16 |   |   |   v.lowlink = min(v.lowlink,w.number);
17 |   |   end
18 |   end
19 |   if v.lowpoint == v.number AND v.lowlink == v.number then
20 |   |   Component c = new strongly connected component;
21 |   |   while s.peek().number ≥ v.number do
22 |   |   |   c.add(s.pop);
23 |   |   end
24 |   |   components.add(c);
25 |   end
26 end
27 for each vertex w in V do
28 |   if w.number == null then
29 |   |   STRONGCONNECT(w);
30 |   end
31 end
32 return components;
```

in its current form will not identify the problematic edges. To solve this problem, the original road network graph was converted into its edge dual, in which verticies are transformed into edges, and edges are transformed into verticies. This method has also been employed by

Demšar et.al. to convert road networks into a more convenient form for analysis [15]. The dual graph can then be processed by Tarjan's algorithm. This will produce multiple lists of components with the largest component being the overwhelming majority of the network that will be used by the simulator. All of the smaller components represent edges from the original graph which are not strongly connected, and which must be removed.

```
<edge id="-15377249#12" from="152380710" to="152397418" name="McLendon Road_1861" priority="7" type="highway.secondary">
    <lane id="-15377249#12_0" index="0" disallow="tram rail_urban rail rail_electric ship"
    speed="27.78" length="90.10" shape="8546.17,3105.66 8634.86,3121.52"/>
    <lane id="-15377249#12_1" index="1" disallow="tram rail_urban rail rail_electric ship"
    speed="27.78" length="90.10" shape="8545.59,3108.91 8634.28,3124.77"/>
</edge>
<edge id="-15377249#14" from="2566985541" to="2566985559" name="McLendon Road_1861" priority="7" type="highway.secondary">
    <lane id="-15377249#14_0" index="0" disallow="tram rail_urban rail rail_electric ship"
    speed="27.78" length="28.53" shape="7605.97,2909.13 7633.72,2915.74"/>
    <lane id="-15377249#14_1" index="1" disallow="tram rail_urban rail rail_electric ship"
    speed="27.78" length="28.53" shape="7605.20,2912.34 7632.95,2918.95"/>
</edge>
```

FIGURE 3.11. An example network file for Rockwall County, TX.

Once the weakly connected components are discovered, NETCONVERT is executed a second time with the specification to remove those particular road segments. Figure 3.11 shows an example of how the edge data is structured in the network file. This produces a strongly connected road network on which routing can now be automatically performed and guarantee a round-trip route fpr a census block group to its associated POD.

Once the road network has been generated, it may need to be adjusted based on the scaling factor. Because a road segment has a finite limit as to how many cars may physically be on it at any moment, in order to have a single agent represent more than one individual it was necessary that the speed of each road segment be scaled accordingly. Equation 3.9 describes the process of calculating the new speed for any given road segment, v', based on that segment's length, l, and speed, v, the average car space, κ, and the scaling factor, α. First, κ is calculated by adding the average length of a car, for which SUMO uses 5 meters as the default, and a minimum gap that each car keeps with the car in front of it, for which SUMO defaults to 2.5 meters. In order to calculate v', it needs to be determined if α will cause a single agent to represent more cars than could physically be on the road at one time. If α is less than or equal to the capacity of the road segment, $\frac{l}{\kappa}$, then v' is equal to v. If

α is greater than the capacity of the road segment, then v' is equal to v multiplied by the capacity over α. Thus, this equation provides the same throughput on a road segment for any agent scaling.

$$v' = \begin{cases} \frac{1}{\alpha} * \frac{l}{\kappa} * v, & \text{if } \alpha > \frac{l}{\kappa} \\ v, & \text{otherwise} \end{cases} \tag{3.9}$$

After the road network has been scaled, census block groups and PODs need to be assigned road segments before routing can be performed. To do this, the road segment GIS information needs to be transfered to the database to make use of the PostGIS functionality. This can be achieved by scraping the network file road segment data, converting that data into well-known text (WKT) linestrings, and then uploading those to the database.

3.3.4. Generating BERPs and Simulation Files

After the road network data is loaded into the PostgreSQL database along with the RE-PLAN response plan data, PostGIS queries can then be executed to assign road segments to each POD and GSU. First, the centroid for each GSU must be computed, as it will then be used to assign road network edges to the GSU as exit and entry points for vehicles of said GSU. The centroid can be calculated using the PostGIS ST_Centroid function. This centroid is then compared against all road segment edge geometries to minimize the distance between the GSU and road segment. The process is repeated for the assignment of road segments to POD locations, but uses lanes instead of edges. That information is used, along with the catchment area assignments and census block group populations, to generate a BERP in VERPETS.

With the BERP generated, the remaining files necessary to execute the simulation can be generated. In order to execute a simulation in SUMO, several files structured in a proprietary XML format are required. Figure 3.12 shows a high-level depiction of the data flow and file creation necessary to generate and execute a simulation. One of the first files needed is a network file, as seen in Figure 3.11, and the creation of which was discussed

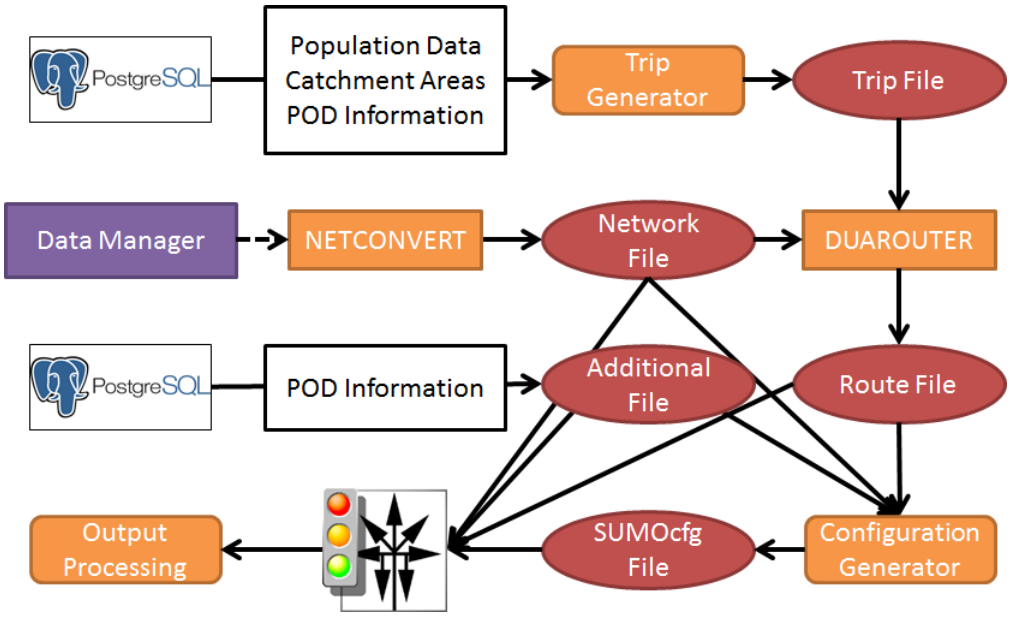


FIGURE 3.12. An overall depiction of data flow and file creation necessary to execute a simulation.

at length in the previous section. Next, a trip file defines the major points a vehicle must traverse during its trip to obtain medication, as exemplified in Figure 3.13. The three major points in this simulation are the edge a vehicle is inserted on (*from*), the edge its POD is located at (*via*), and the edge it leaves from (*to*). Consequently, the *from* and *to* edges are the same as an individual will exit and enter their census block group through the same road segment. Also, each vehicle must have a trip defined for it in order to generate a route for that vehicle.

```
<?xml version="1.0"?>

<trips>
    <trip id="1" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="2" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="3" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="4" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="5" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="6" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="7" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="8" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="9" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="10" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="11" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
    <trip id="12" depart="0.00" from="386890767#1" to="386890767#1" via="15379345#5"/>
```

FIGURE 3.13. An example trip file for Rockwall County, TX.

Additionally, the departure time of a vehicle (depart) can be altered to simulate different vehicles leaving at different times. Effectively, the simulator loads a car into the simulation at its departure time, where it waits in a queue, called the insertion-backlog, to be inserted onto the road network. Once inserted the vehicle follows its route, which will be discussed next, to its POD location and then back to the road segment it was inserted on. Upon reaching the final edge in the route, the car is removed from the simulation and marked as arrived.

The vehicle release interval, as describe in Table 3.5, controls how often cars are loaded into the insertion-backlog. If the vehicle release interval is set to 0, all cars are loaded into the simulation and their respective backlogs at the beginning of the simulation. Given a sufficiently large number of agents, this can cause significant slow-down in simulator performance and is not recommended without using scaling or the cut-off. Additionally, even though all agents are loaded into the simulation, they can only be loaded onto a road segment as space is available. Therefore, there is usually a few second delay insertions of agents onto the road network. This leads to a repeated burst pattern as shown in Figure 3.14. Each peak represents all census block groups releasing a single car, and then needing to wait two seconds before a sufficient amount of space on the road is available to insert the next. This is a result of agents starting at a speed of 0 kmph when they enter the road network, and then accelerating up to the maximum speed for that road segment over time. It should be noted that not all of the census block groups may remain in sync as any shared usage of the road network may lead to insufficient room at the insertion segment for a census block group.

If scaling is applied to the simulation, then a rounding strategy may need to be applied to account for situation in which the population of a census block group do not evenly divide by the scaling factor. The rounding configuration, as describe in Table 3.5, can be set to four different strategies: ceiling, floor, rounding to the nearest whole number, and no rounding. If ceiling is used, whenever a census block group has an insufficient number of people to fully form a new agent, population is rounded up to be evenly divisible by the

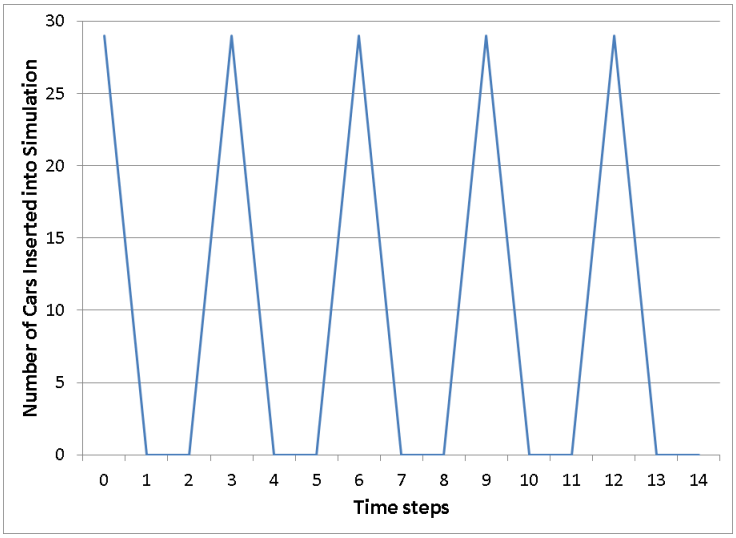


FIGURE 3.14. An example of insertion rates of agents into the network when all agents are loaded into the insertion-backlog.

scaling factor and form the agent. If floor is used, whenever a census block group has an insufficient number of people to fully form a new agent, the agent is not formed and the population is effectively rounded down to be evenly divisible by the scaling factor. Closest rounds the population to the nearest population that would be evenly divisible by the scaling factor. Finally, None does not round the population at all and allows for agents that less than the scaling factor number of people. In this case, the processing time for a partial agent is equal to the average processing time multiplied by the number of individuals the partial agent is representing. Table 3.6 provides an example of the effects of each strategy on total population of a catchment. It should be noted that each of these strategies, except None, has a compounding error because each census block is rounded. Experiments in chapter 4 will provide greater detail on how these strategies affect the resulting completion time of simulation activation.

After the trip and network files have been created, the edge-by-edge route for each vehicle can be generated. As shown in Figure 3.12, both of these files are taken as inputs to DUAROUTER, a routing script that comes with SUMO. The script will then read the trip file, and for every trip calculate an optimal route along the road network from the starting point, to the POD, and back to the ending point. DUAROUTER comes with multiple routing

Catchment Area	Actual Population	Ceiling	Floor	Closest	None
1	9,823	10,200	9,300	9,900	9,823
2	4,718	4,900	4,500	4,700	4,718
3	10,136	10,700	9,400	10,100	10,136
4	3,780	4,000	3,600	3,800	3,780

TABLE 3.6. A list of total populations for each catchment area in Rockwall County before and after different rounding strategies are applied.

algorithm options, such as Dijkstra's or A*, and Dijkstra's was chosen for this system. An example of a generated route file can be found in Figure 3.15.

```
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http:,
    <vehicle id="1" depart="0.00">
        <route edges="386890767#1 386890767#2 386890767#3 386890767#4 386890767#5 386890767#6 3868
    </vehicle>
    <vehicle id="2" depart="0.00">
        <route edges="386890767#1 386890767#2 386890767#3 386890767#4 386890767#5 386890767#6 3868
    </vehicle>
    <vehicle id="3" depart="0.00">
        <route edges="386890767#15 386890767#16 386890767#17 386890767#18 386890767#19 386890767#2
    </vehicle>
    <vehicle id="4" depart="0.00">
        <route edges="144977371#16 -144977371#16 -144977371#15 -144977371#14 -144977371#13 -144977
    </vehicle>
    <vehicle id="5" depart="0.00">
        <route edges="1640051986#3 386890767#0 386890767#1 386890767#2 386890767#3 386890767#4 3868
    </vehicle>
```

FIGURE 3.15. An example route file for Rockwall County, TX.

However, DUAROUTER does not generate all of the information necessary to properly collect output from a simulation. Therefore, a post-processing step is performed on the route file. Figure 3.16, shows the final route file that will be used by SUMO. The first piece of information that must be added is the vehicle definition, or vType. This allows the creation of different styles of drivers with different max speeds, acceleration times, or car sizes. Once a vType is defined, it must be applied to each vehicle by setting the vehicle type to the vType id. This vType id is also used in the additional file, as will be described later.

The second addition to the route file is the stop definition. In order to receive treatment at a POD location, an individual must spend a certain amount of time in processing at the POD. This processing time accounts for activities such as filling out and submitting forms, standing in line, answering questions, and actually receiving the medication. Thus, a

```
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http:

    <vType  id="Car" lcStrategic="100.0" lcSpeedGain="100.0" lcKeepRight="0.0"/>

    <vehicle id="1" depart="0.00" type="Car">
        <route edges="386890767#1 386890767#2 386890767#3 386890767#4 386890767#5 386890767#6 386
        <stop parkingArea="1" duration="180" />
    </vehicle>
    <vehicle id="2" depart="0.00" type="Car">
        <route edges="386890767#1 386890767#2 386890767#3 386890767#4 386890767#5 386890767#6 386
        <stop parkingArea="1" duration="180" />
    </vehicle>
    <vehicle id="3" depart="0.00" type="Car">
        <route edges="386890767#15 386890767#16 386890767#17 386890767#18 386890767#19 386890767#
        <stop parkingArea="1" duration="180" />
    </vehicle>
    <vehicle id="4" depart="0.00" type="Car">
        <route edges="144977371#16 -144977371#16 -144977371#15 -144977371#14 -144977371#13 -14497
        <stop parkingArea="1" duration="180" />
    </vehicle>
    <vehicle id="5" depart="0.00" type="Car">
        <route edges="164051986#3 386890767#0 386890767#1 386890767#2 386890767#3 386890767#4 386
        <stop parkingArea="1" duration="180" />
    </vehicle>
```

FIGURE 3.16. An example refined route file for Rockwall County, TX.

each vehicle will need to spend a user defined duration stopped at the POD. SUMO allows this functionality through a feature known as a stop. In this case, a parkingArea style stop was chosen, as that allows vehicles to leave pull off the road and wait their duration to be processed. This also allows for concurrent processing of multiple vehicles. As each POD has more than one booth which it can use to process individuals, one parking space was added to the lot for each booth to simulate the concurrent processing. Further, if the agents are scaled, their processing time is multiplied by α to more accurately reflect the amount of time the scaled number of individuals the agent represents would take to be processed at the POD.

```
<additional xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/additional_file.xsd">

<parkingArea id="1" lane="15379345#5_0" startPos="10" endPos="26" roadsideCapacity="16" />
<parkingArea id="2" lane="180785430#1_0" startPos="10" endPos="18" roadsideCapacity="8" />
<parkingArea id="3" lane="34503516#2_0" startPos="10" endPos="27" roadsideCapacity="17" />
<parkingArea id="4" lane="225339911#10_0" startPos="10" endPos="17" roadsideCapacity="7" />

<vTypeProbe id="1" type="Car" freq="1" file="-"/>

</additional>
```

FIGURE 3.17. An example additional file for Rockwall County, TX.

Next, the additional file maintains information on the parkingArea for each POD location, as well as vehicle type probe information. Each parkingArea receives a unique id, a lane that it is assigned to, a starting and ending point, and capacity. The capacity denotes how many cars may park in the lot at one time, and is directly proportional to the number of booths at that POD location. This allows for the simulation of concurrent processing of vehicles, just as would happen at a real POD location. The simulated space that the POD occupies is equal to two times capacity of the POD. Thus, endPos for the parkingArea is set at two times the capacity plus the startPos.

Additionally, as mentioned previously, the vehicle type probe, or vTypeProbe, is defined for each vehicle type to allow the system to monitor and report on the location of the vehicle throughout the duration of the simulation. Each probe has a unique id, a vehicle type that it corresponds to, and a frequency for how often data is collected about the vehicle. For this research, only one probe is defined as all cars are the same type, and the frequency of data collection is always set to one to collect the maximum amount of information.

```
<configuration>
    <input>
        <net-file value="Rockwall.net.xml"/>
        <route-files value="Rockwall.route.xml"/>
        <additional-files value="Rockwall.addl.xml"/>
        <time-to-teleport value="-1"/>
    </input>
</configuration>
```

FIGURE 3.18. An example sumocfg file for Rockwall County, TX.

Finally, the SUMO configuration file, or sumocfg, is used to specify the network file, route file, and additional file used in the simulation. An example of this file can be seen in Figure 3.18. Further, additional simulation configurations can be made, such as turning off vehicle teleportation. By default, SUMO assumes that if a car has to wait too long at a location that something has gone wrong in the simulation and teleports it further along its route. Since it is assumed that heavy and longterm congestion will occur along the road to the PODs, the time-to-teleport was set to -1 to disable teleporting.

3.3.5. Static Flow Analysis

Before the simulation was performed some analysis and statistics about the plan and its associated road network can be performed. For example, it is useful to emergency managers and planners to understand how the road network will be utilized during plan activation. This information can be used to take preparatory measures in heavily used locations by potentially assigning police or other emergency personnel to more quickly solve complications that could arise. Thus, a method for performing a network flow analysis on the network using only the census block population and route information was created.

To perform this analysis, first, the routes for the plan must be read in from a route file. Because these files can normally have thousands or tens of thousands of entries, one entry for each vehicle, a faster method was devised. New trip files were generated with one trip per census block group. This will allow for the fast generation and processing of route files, while still providing the necessary routing information. As each route is read in, the population of the census block group is added to the total number of cars that will use each road segment in the route. Additionally, the number of unique PODs and census block groups that use a road segment are tabulated, as this can provide information on route overlaps between catchment areas.

Finally, this information can be used to generate maps of the road network itself. By using GIS software, such as ArcGIS or QGIS, the data in the network shapefile that is generated when the road network is processed can be joined with the road use data using the road segment ID. This allows maps, such as Figure 3.19 to be constructed, which can show where high rates of road usage could be. In this case, road segments which are more red have a higher usage, whereas those that are more green have little usage.

3.3.6. Validation Simulation

Once all of the necessary files have been generated, a simulation can be executed. This is performed by calling the SUMO executable and providing it the SUMO configuration file. However, initial trials with SUMO showed two major issues with simulation execution. The first dealt with the output generated by the experiment. In order to accurately determine

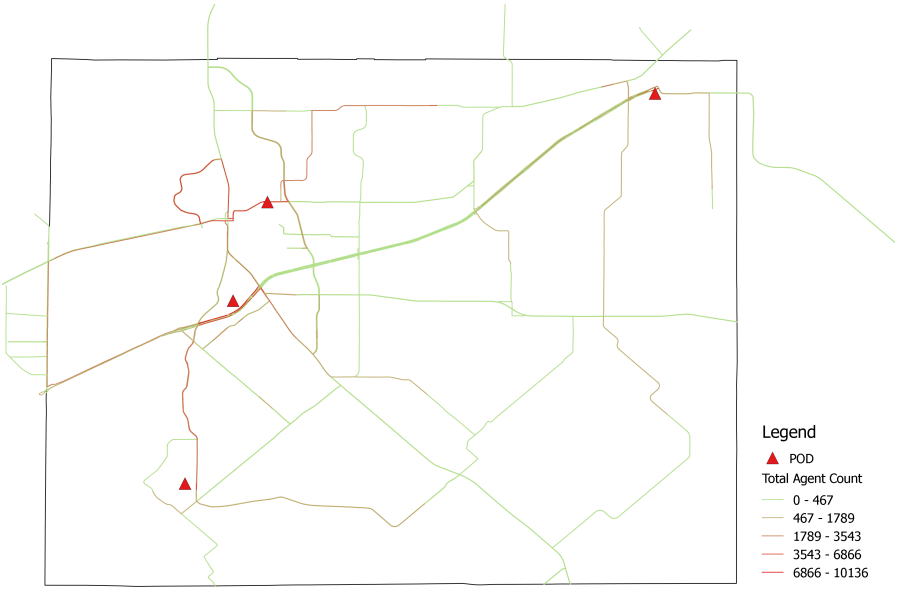


FIGURE 3.19. An example road use map for a Rockwall county plan.

how vehicles were traveling through the network over time, and determine when vehicles and reached and left the PODs, the position of each vehicle needed to be analyzed every time step. Due to the number of cars that were concurrently in the simulation, if the results were stored, a single simulation output file could quickly balloon to hundreds of gigabytes of data. This was untenable in terms of both storage and post-simulation processing. The second problem dealt with how long the simulations took to execute. SUMO is, unfortunately, single-threaded and thus when thousands, tens of thousands, or hundreds of thousands of vehicles are present in the simulation it causes significant slowdowns in simulation. Early simulations were allowed to process for multiple days with no clear sign of when they would complete. Again, this was not feasible since validating a multiple similar plans require multiple simulations to be executed. Conveniently, both of these problems can be solved by retaining only the necessary agent position data and using either a halting strategy or scaling.

To allow VERPETS to read SUMO's output during simulation execution the simulation itself must be configured correctly, and a tool to sample the output must be created. To allow for easier collection of data, the simulation is set to output all output, including

warnings and errors, to standard out. This is achieved by both setting the file attribute to "-" in the vTypeProve, as seen in Figure 3.17, and by configuring the Java ProcessBuilder library to redirect standard error output to standard out. This combined stream of data can now be read, provided that there is a something listening to standard out.

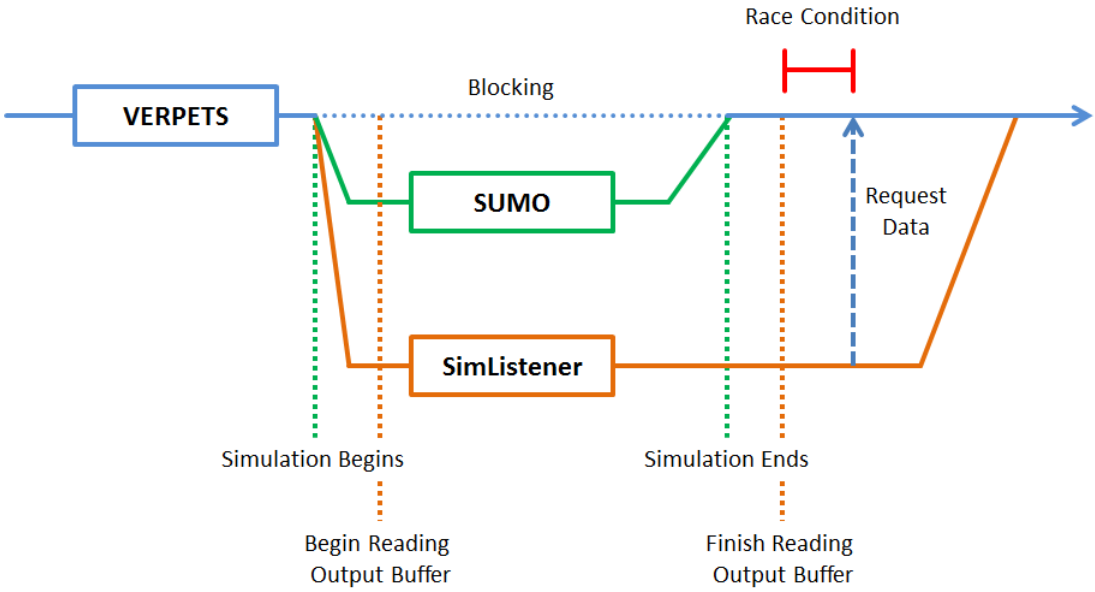


FIGURE 3.20. A timing diagram of simulation execution.

Thus, the Simulation Listener, or SimListener, was designed to execute as a separate process and collect data from SUMO during execution. Figure 3.20 presents a timing diagram of the different threads of execution. Each time a simulation is executed, a new SimListener is also generated and given a pointer to the simulation's process thread. This allows the SimListener to directly read from the simulation's output. As the SUMO simulates each time step, the SimListener will analyze each car within the timestep and determine if it is in one of three states: before the POD, at the POD, after the POD. All agents start off in the before the POD state, which indicated they have not yet reached the road segment their POD is located on. Once they do reach the segment, they are marked as at the POD. They will remain in that state until they have left the POD and the road segment their POD is on. Once that happens, they are marked as after the POD. If no advanced halting strategy is utilized, then the simulation is ended once the last agent in the simulation is marked as

after the POD. By only maintaining the states of each agent, this represents and enormous saving in storage capacity, as the overwhelming majority of the data output by SUMO is discarded as unnecessary.

While halting the simulation once the last agent has been marked after the POD may work with sufficiently scaled agents, this strategy could incur significant real-world time costs with a 1:1 scaling. Thus, a more advanced halting condition was needed. As was previously mentioned, if congestion or runaway congestion are allowed to occur, then after a period of time, travel time does not impact plan completion. Thus, once either of those occurs, a simple rate based calculation can be performed to estimate plan completion time. To calculate this, first the total number of unprocessed vehicles must be calculated. This can be performed using Equation 3.10, where *Total Vehicles* is the the count of all households in the county, *Total Processed Vehicles* is the the number of vehicles processed before the simulation was halted, and *Total Exited Vehicles* is the number of vehicles that have been processed and exited the simulation. Once the number of unprocessed vehicles has been calculated, the BERP completion time can be calculated using Equation 3.11. In this Equation, *Final Timestep* is the last timestep simulated by the simulator, *Average Processing Time* is the average processing time for all booths, *Total Booths* is the total number of booths in the BERP, and *Unprocessed Vehicles* is the result from Equation 3.10. These Equation effectively estimate the completion time due to the processing rate becoming, as previously shown, the rate limiting step of the simulation.

(3.10)

$$Unprocessed\ Vehicles = Total\ Vehicles - (Total\ Processed\ Vehicles\ - Total\ Exited\ Vehicles)$$

(3.11)

$$BERP\ Completion\ Time = Final\ Timestep + \frac{Average\ Processing\ Time}{Total\ Booths} * Unprocessed\ Vehicles$$

However, before these equations can be used, it must be determined when a POD

has reached congestion or runaway congestion. Preliminary experiments were performed to test whether if some standard time could be used to halt the simulation and then estimate the final completion time. Using SUMO's GUI, a visual inspection of when a POD reached runaway congestion was performed. In this case, a POD was said to have achieved runaway congestion if the number of agents waiting to enter the POD was one plus the number of booths. Once a POD reached that point, the elapsed simulation, in seconds, was recorded. Table TODO shows the results of these preliminary experiments. It can be concluded from the table, that setting a standard cutoff time of 1800 seconds, or one hour, always allows for a POD to reach runaway congestion and thus was used in the implementation of VERPETS.

Finally, once the SimListener reaches halting condition, it will end the simulation. Because the SimListener has access to the simulation's process thread in Java, it simply calls the destroy() function for that process. As shown in Figure 3.20, VERPETS blocks while the simulation is executing and then requests the processed data from the SimListener once the simulation is ended. This was determined to cause a race condition between VERPETS and its SimListener, and would occasionally result in the SimListener providing incomplete data. The race condition was remedied by adding in a readyToRead flag, that was set by the SimListener when it had finished reading the last of the simulation's output. The SimListener can then estimate completion time of the BERP and provide output to the used.

3.3.7. Reporting Results

The results for each simulation are simply reported to standard out. This includes a list of when each POD completed processing its entire population, which POD took the longest which provides the final completion time for plan activation, and whether or not the plan is feasible according to the user defined time limit.

CHAPTER 4

EXPERIMENTS AND RESULTS

To test the effectiveness of VERPETS ability to evaluate emergency response plans, multiple experiments were performed using different emergency response plans. In this case, six plans were developed to examine VERPETS performance, and will be described here. First, a traffic flow analysis was performed on each plan to both generate data used by the scheduling analysis, and to generate maps allowing a better understanding of the distribution of traffic across the road network. Next, VERPETS's ability to validate each plan was examined, and compared to analysis available via RE-PLAN. Due to the memory and processing requirements of large simulations, multiple methodologies for reducing the computational intensity were examined. These experiments, and their results, will be discussed in this chapter.

4.1. Response Plans

As stated, in order to evaluate VERPETS's ability to validate biological emergency response plans (BERPs), it was necessary to generate multiple BERPs. This was performed by using RE-PLAN to develop plans for two different sized counties in Texas: Rockwall and Denton county. Table 4.1 shows the statistics of the two counties selected.

County	**Land Area**	**Population**	**Number of Households**	**Population Density**	**Number of Census Block Groups**
Rockwall	127 mi^2	85,536	28,457	634/mi^2	30
Denton	878 mi^2	731,851	275,275	834/mi^2	378

TABLE 4.1. County details for each county used in this research.

Rockwall county, the smaller of the two and also the smallest county in the state, provides an example of a county with a small land area and low population density. These features make Rockwall desirable to initially examine as it takes little time to process and simulate the county, and thus makes system testing expedient. The county contains the cities of Fate, Heath, McClendon-Chisolm, Mobile City, Rowlett, Royse City, and its county

seat is in the city of Rockwall. The major highway in Rockwall county is I-30, which travels from the West to the North-East. Figure 4.1 represents the census block group division of the county on the left and shows the same division with major roads overlaid in red on the right.

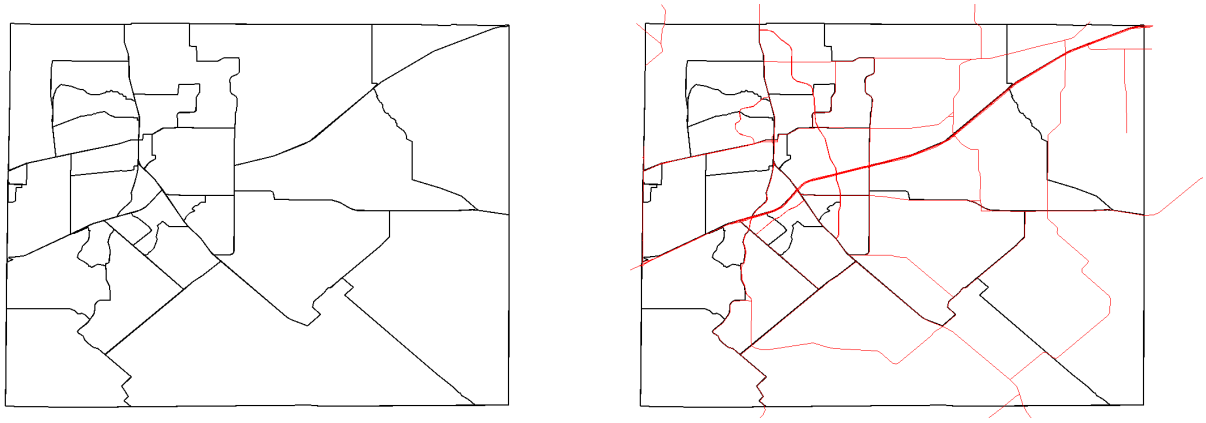


FIGURE 4.1. Left: Census block group divisions for Rockwall county. Right: Census block group divisions for Rockwall county with major roads overlaid.

While exhibiting a similar population density to Rockwall county, Denton county covers a land area almost nine times as large. This large increase in population also results in a more than ten-fold increase in the number of census block groups. To support this population, Denton contains a far more robust road network which includes two major interstate highways, I-35E, and I-35W, and three major highways US 77, 377, and 380. The county includes populations from several major and minor cities including Plano, Frisco, Carrollton, Lewisville, Denton, Corinth, and Flowermound. Figure 4.2 shows the census block group division of the county on the left and shows the same division with major roads overlaid in red on the right.

In order to simulate a BERP for each of these counties, a number of emergency response plans were created. Table 4.2 describes the six plans, and Figures 4.3 and 4.4 show the catchment areas and POD locations for each of the plans. Rockwall only has a single map because the catchment areas did not need to be recreated for each new plan. This is due to the fact that altering the number of booths did not change the number of required PODs, and thus the population did not need to be repartitioned.

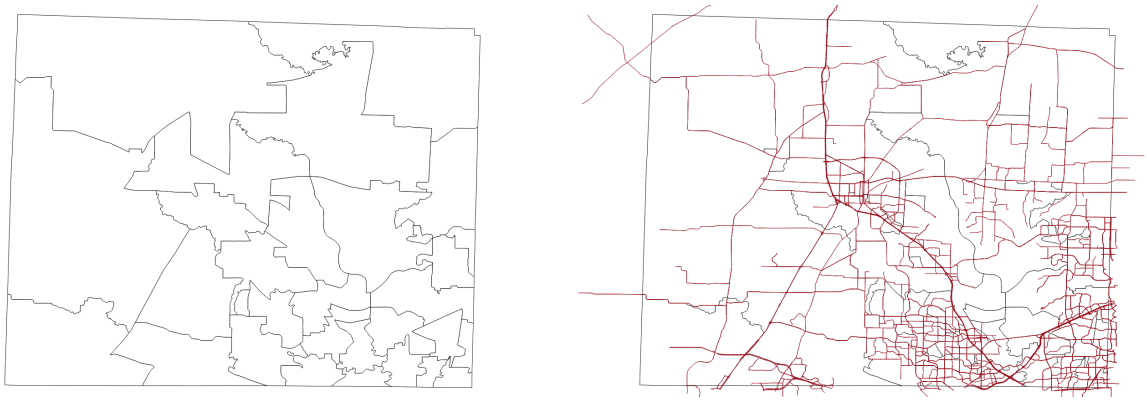


FIGURE 4.2. Left: Census block group divisions for Denton county. Right: Census block group divisions for Denton county with major roads overlaid.

County	Response Plan Name	Number of PODs	Total Number of Booths	Processing Time (sec)
Rockwall	rockwall_1	4	48	180
	rockwall_2	4	35	180
	rockwall_3	4	40	180
Denton	denton_1	29	429	180
	denton_2	22	322	180
	denton_3	24	358	180

TABLE 4.2. A list of the response plans used for evaluating VERPETS.

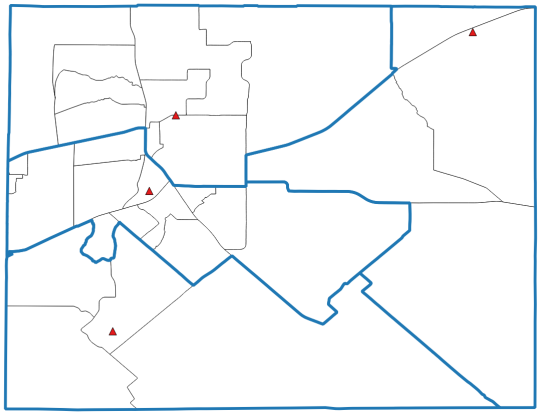


FIGURE 4.3. The catchment areas and POD locations for rockwall_1, rockwall_2, and rockwall_3.

All of these plans assume a 36 hour deadline for the treatment of the entire population, and were constructed to examine the four different possible outcomes of plan development:

- Theoretically Feasible and Realistically Feasible

(A) denton_1 (B) denton_2

(C) denton_3

FIGURE 4.4. The catchment areas and POD locations for denton_1, denton_2, and denton_3.

- Theoretically Feasible and Realistically Infeasible
- Theoretically Infeasible and Realistically Feasible
- Theoretically Infeasible and Realistically Infeasible

A plan is said to be theoretically feasible if the completion time, as evaluated by Equation 4.1, is less than or equal to the time limit. It is deemed theoretically infeasible otherwise. A plan is said to be realistically feasible if the completion time, as determined by plan activation simulation, is less than or equal to the time limit. It is deemed realistically infeasible otherwise. In this case, only one plan was developed for each county that was theoretically infeasible, as Equation 4.1 provides the theoretical minimum completion time. Thus, if a plan is theoretically infeasible, it is impossible for it to be realistically feasible.

$$Completion\ Time = \frac{\phi}{Total\ Number\ of\ Booths * A} \tag{4.1}$$

The first plan for each county was designed to have a completion time around 30 hours, which is well below the deadline. This represents a plan that was both theoretically and realistically feasible. By setting the planned completion time at 30 hours, this ensured that even with some possible traffic delays, the plan would complete before the 36 hour limit. The second plan for each county was designed to have a completion time for 40 hours, thus causing it to exceed the deadline by a great margin. This represents a plan that was both theoretically and realistically infeasible, due to the insufficient number of resources allocated to the plan. The third plan for each county was designed to have a completion time as close to, but below, the 36 hour time limit. This plan was designed to be theoretically feasible, but potentially could be infeasible when activated due to traffic dynamics and booth distribution.

4.2. Static Traffic Flow Analysis

Once a response plan was generated an analysis of the traffic flow across road network could be performed. Because SUMO generated a route for each census block group before simulation, the number of individuals on each road segment could be calculated. Thus, an analysis of which road segments had the largest number of agents cross them during plan activation could be performed. Figure 4.5 showed this analysis for rockwall_1, rockwall_2, and rockwall_3. The Jenks natural breaks classification method [33] was used to generate the five classifications.

Due to the fact that the Denton plans had different numbers of PODs, and thus catchment areas, a unique, albeit similar, analysis was produced for each one. Figure 4.6 shows an analysis for denton_1, Figure 4.7 shows an analysis for denton_2, and Figure 4.8 shows an analysis for denton_3.

As can be seen in Figures 4.5, 4.6, 4.7, 4.8, some of the PODs were relatively close together, which could have allowed for some census block groups from one POD to have used the same road network segments as census block groups from another POD. In order

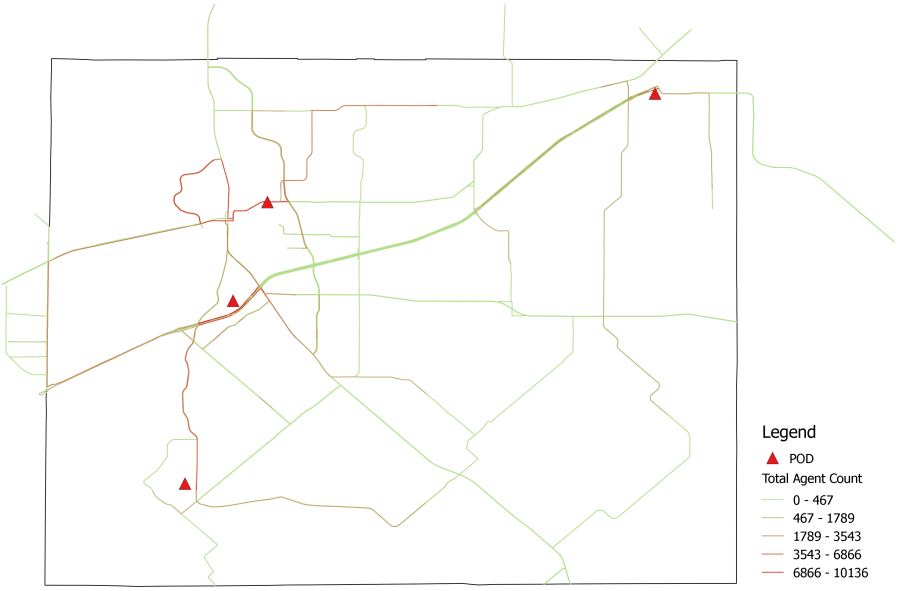


FIGURE 4.5. The load on the road network for Rockwall county during plan activation.

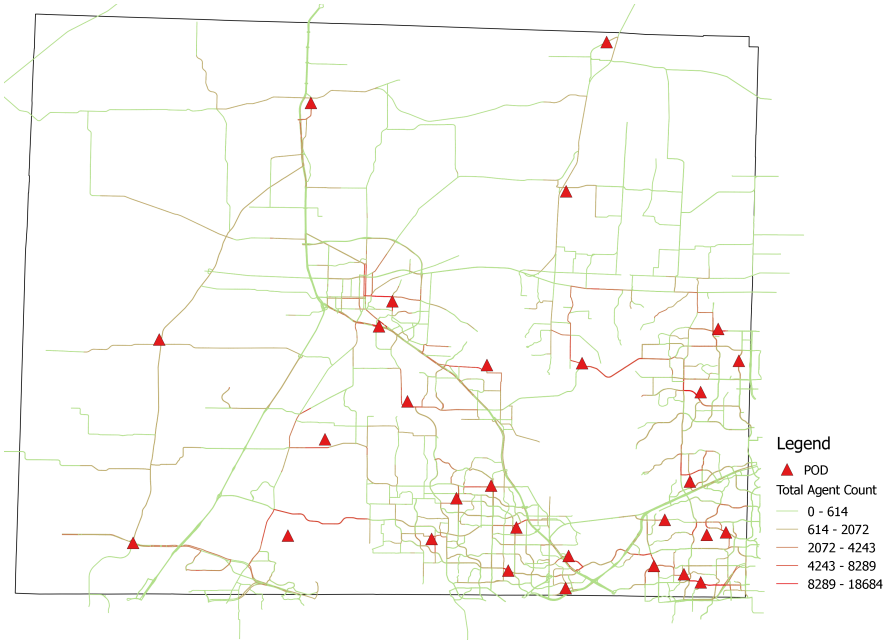


FIGURE 4.6. The load on the road network for Denton county during plan activation for the plan denton_1.

to determine the extent of this potential overlap, further analysis was performed using the static flow analysis.

To calculate the overlap between different PODs' usage of the road network, the route

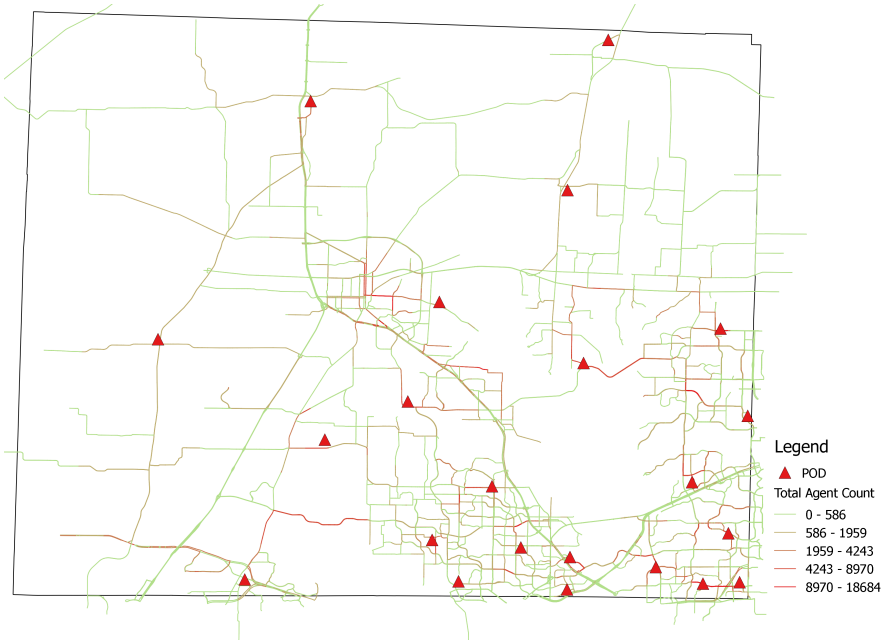


FIGURE 4.7. The load on the road network for Denton county during plan activation for the plan denton_2.

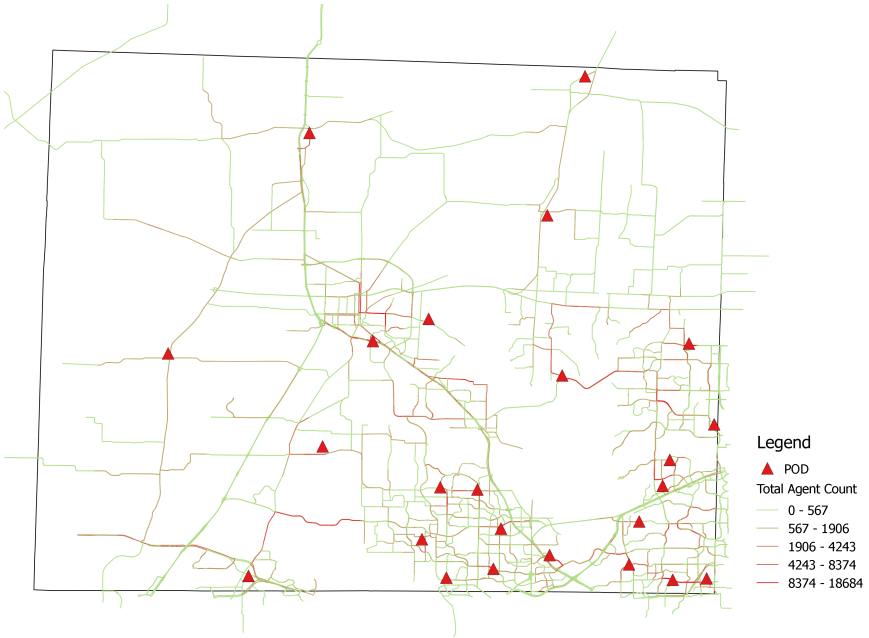


FIGURE 4.8. The load on the road network for Denton county during plan activation for the plan denton_3.

from each census block group to its POD were first generated. Next, based on those routes, the number of PODs using each road segment were counted. This was used to determine the number of road segments that were unused, used by one or more PODs, and used by more

than one POD. Figure 4.9 displays this count for rockwall_1, rockwall_2, and rockwall_3. In these plans, less than half of the entire road network was used for routing, and only roughly 7% of what was used was used by census block groups from more than one POD. Therefore, for Rockwall county plans the overlap was minimal.

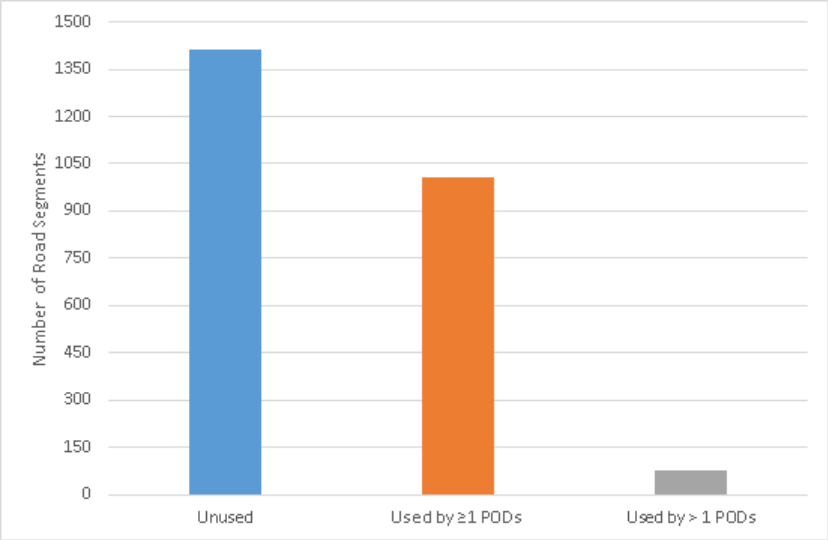


FIGURE 4.9. The number of road segments used for routing rockwall_1, rockwall_2, and rockwall_3

Figure 4.10 shows similar outcomes for the three Denton county plans. Again, less than half of the total road network for Denton was used for routing. Additionally, only roughly 6% of the road segments for denton_1, roughly 5% of the road segments for denton_2, and roughly 6% of the road segments for denton_3 were used by census block groups from more than one POD. Again, it can be concluded that overlap on the road network was minimal between the PODs in these plans.

4.3. Response Plan Validation through Simulation

As explained in Chapter 3, there were multiple possible attributes which could be configured for a simulation. In order to better understand how these affect the outcome of a simulation, and to determine which combinations leads to the best results and software performance, several experiments were performed. All experiments were performed on a Dell Optiplex 980 with an Intel Core i7 CPU, 12 GB of RAM using Windows 7.

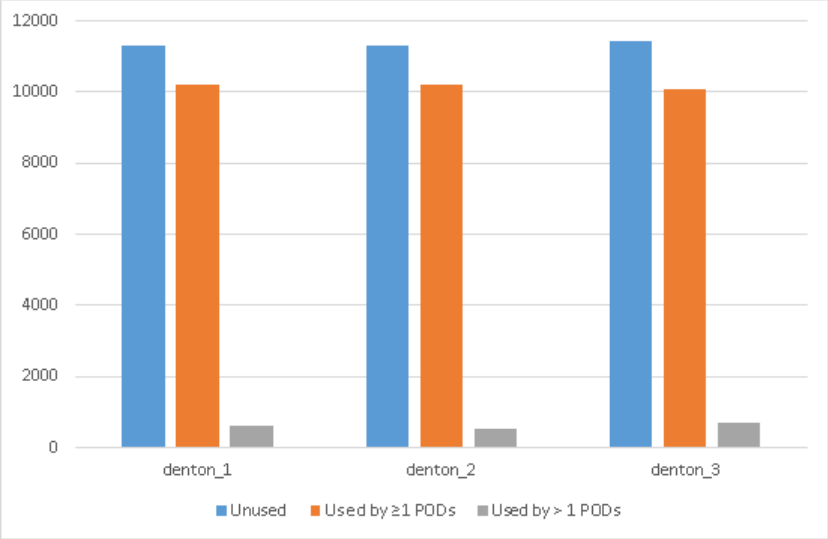


FIGURE 4.10. The number of road segments used for routing denton_1, denton_2, and denton_3

4.3.1. Whole Plan vs Catchment Areas

During preliminary testing, it was determined that SUMO suffered a significant performance drop as more agents entered the simulation. One of the strategies to increase performance and decrease the time to simulation completion was to break the plan into its component catchment areas and simulate each catchment area in parallel. Due to these catchment areas being formed by performing a closest POD partitioning, overlap between them was minimized, as shown previously. However, because of the possibility of perturbations caused by these overlaps, experiments were performed to analyze the difference between simulating all of the agents in a plan at the same time, and simulating the catchment areas in parallel. Additionally, this potential difference was examined with respect to the two main strategies for simulating a plan, using scaling and using a time limit.

In these experiments, the Rockwall plans were used with a processing time of 180 seconds. As will be discussed later, in order to reduce agent load on the simulator, two main strategies were used when executing simulations. The first was scaling, in which a single agent in the simulation represents more than one individual of the population. Scalings affects the plans by decreasing the maximum road network speeds to maintain the original traffic flow through the network, and increasing the processing time at the POD for an agent.

The second strategy is to halt the simulation after some time and calculate the predicted end time based on how many agents have already been processed and how many still must be processed. For these experiments, when using scaling, a ratio of 100 individuals per 1 agent was used, and a time limit of one hour was used. Figure 4.11 shows the completion times for each of the three response plans using a time limit and 4.12 shows the completion times for each of the three response plans using scaling. The maximum and minimum values for the simulations are represented by the black bars. As the figures show, there was a negligible difference between the completion times when simulating all catchment areas of a plan together as one simulation, as compared to simulating each catchment area individually in parallel. Additionally, as Table 4.3 shows, the variation around the mean was very low across the simulations used to generate the average. Due to these factors, the similar completion times, the variation, and the maximum and minimum values close to the average, it can be concluded that simulating the catchment areas in parallel is nearly equivalent to simulating the catchment areas in one simulation. This is beneficial due to the significant difference in run-times for each method. On average, simulating the catchment areas in parallel took roughly three minutes on average, whereas simulating the catchment areas in a single simulation took roughly seven minutes on average. This represents a considerably time savings especially when simulating large counties with considerably more agents.

Response Plan Name	**Separate w/ Scaling**	**Whole w/ Scaling**	**Separate w/ Time Limit**	**Whole w/ Time Limit**
rockwall_1	0.06	0.11	0.00	0.00
rockwall_2	0.11	0.09	0.00	0.00
rockwall_3	0.00	0.00	0.00	0.00

TABLE 4.3. The variation for each of the experiments in 4.11 and 4.12.

4.3.2. Rounding Strategies

Another strategy to reduce overall system run-time was to scale how many individuals a single agent represents. However, the populations of each census block group may not evenly divide by the scaling factor. Thus, four strategies for how to handle potentially partial agents was investigated. The *ceiling* strategy assumes a partial agent represents a

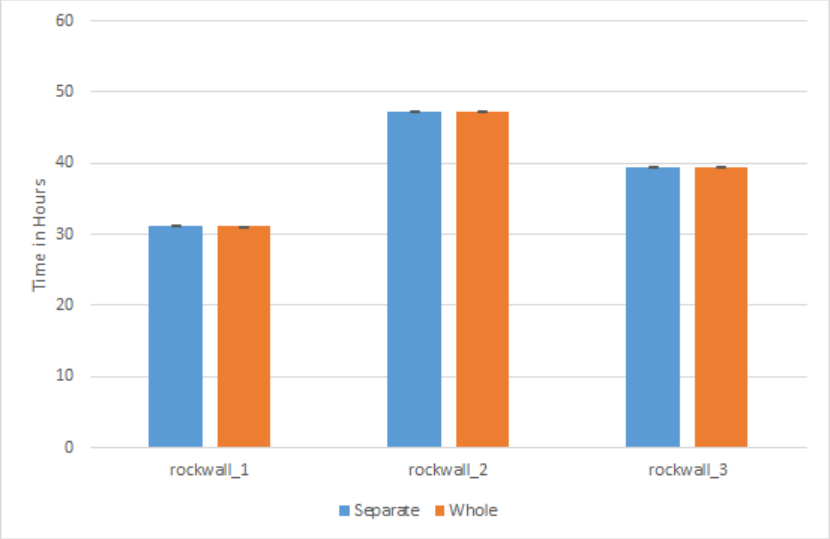


FIGURE 4.11. The effects of simulating all catchment areas at once (whole) vs simulating catchment areas separately in parallel while using a one hour time limit.

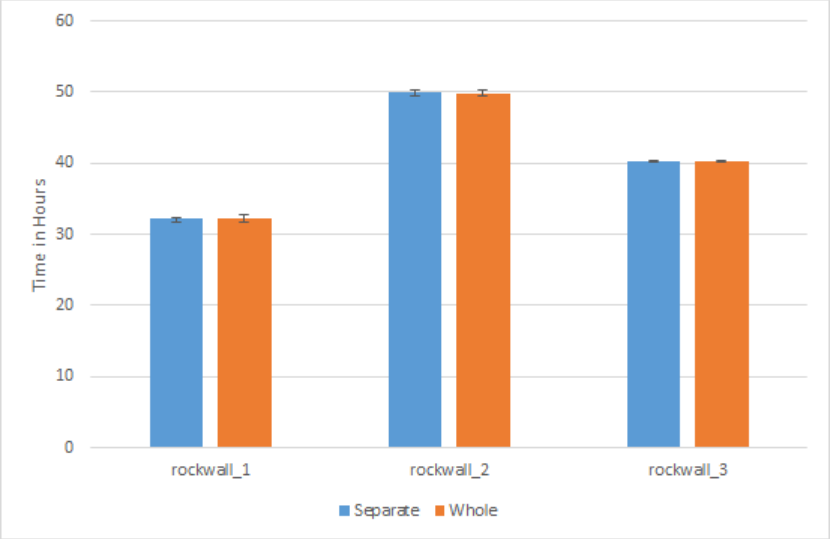


FIGURE 4.12. The effects of simulating all catchment areas at once (whole) vs simulating catchment areas separately in parallel while using scaling.

whole agent. The *floor* strategy ignores any partial agents. The *closest* strategy assumes a partial agent represents a whole agent if it is as least 50% of the represented individuals, or ignores it otherwise. Finally, the *none* strategy allows a partial agent to remain a partial agent. Figure 4.13 shows the affects on the population when these different strategies with a 100 to 1 scaling.

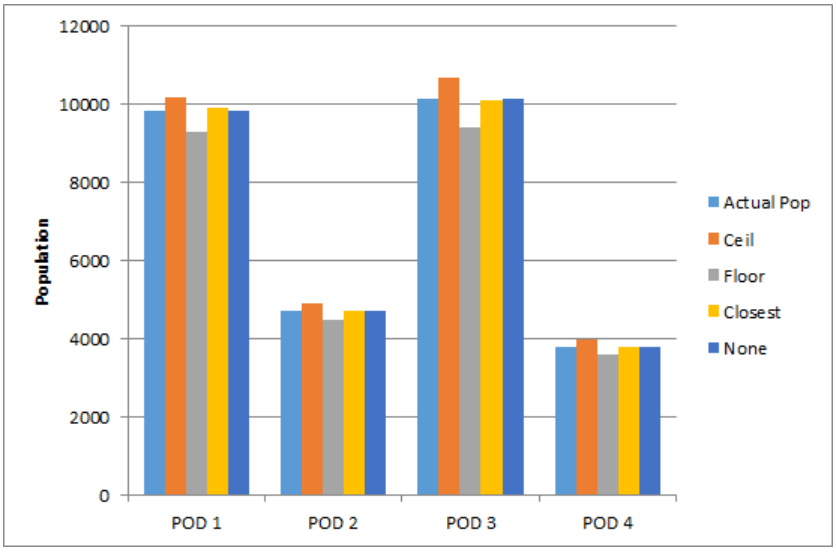


FIGURE 4.13. The resulting populations after rounding strategies were applied.

As expected, using *ceiling* always overestimated the population, *floor* underestimated the population, using *closest* provided the closest estimation, and using no rounding resulted in the actual population. The main reason why *ceiling* and *floor* are typically farther away from the actual population than *closest*, has to do with a compounding error effect. Utilizing one of the rounding strategies rounded the population for each census block group, not the total population. Thus, there was repeated under or over estimation of the actual population which lead to the larger difference. Utilizing a *closest* rounding, helped to balance the error between the two previous strategies, but could be subject to over or under estimation depending of the distribution of the population across the census block groups.

The affects of the different rounding strategies were also tested on completion time of the simulation. Figure 4.14 shows the affects on the population when these different strategies with a 100 to 1 scaling, a processing time of 180 seconds, by allowing all agents to be released at once.

In this case, the completion times of each POD varied greatly depending on which rounding strategy was used. This significant difference can be attributed to the fact that, when using a rounding strategy, a single agent's processing time is multiplied by the scaling factor. In this case, a car would need to be processed at a booth for 18,000 seconds, or five hours. Thus, if there were a few additional cars for a POD to process when using

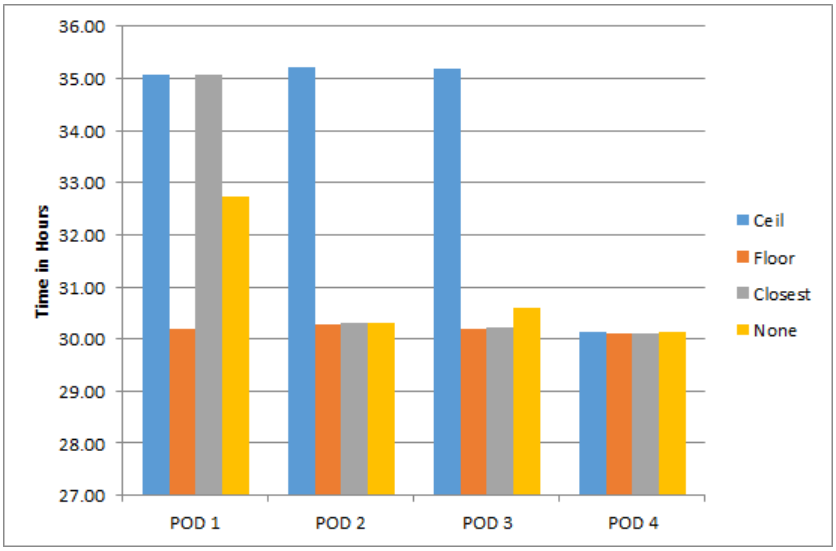


FIGURE 4.14. Simulation completion times resulting from rounding strategies.

ceiling, as opposed to *floor*, the plan could take an extra five hours to execute, which is exactly the outcome POD 1,2 and 3 experienced in Figure 4.14. Conversely, if fewer cars were simulated than actually exist, the plan execution could have taken five fewer hours. Even the *closest* rounding strategy can be affected by this problem, so it was determined that using no rounding strategy, and allowing partial agents was the preferable to the error caused by a rounding strategy.

Additionally, even though using no rounding strategy does not accurately simulate the travel time of a partial agent, this underestimation of travel time is negated by the congestion at the POD. This is the case because the partial agent is always the last agent to be released and thus would be the final agent in the queue. Further, because vehicles are now being processed in the POD for such a scaled up duration, most agents are able to reach and queue up at the POD before the first agent has completed processing. Thus, the partial agent's travel time would not factor into the simulation at all, and is therefore the best rounding strategy to use for these simulation.

4.3.3. Scaling Increments

In addition to testing what effect the rounding strategies had on scaling, determining an appropriate scaling ratio could help in achieving improved performance. However, because

scaling affects the road network itself, experiments needed to be performed to examine the impact of different ratios on simulation completion time. Thus, simulations were performed at 25:1, 50:1, 75:1 100:1, 125:1, 150:1, 175:1, and 200:1. Each simulation was used a 180 second processing time, all vehicles were released at once, and used partial agents. Figure 4.15 shows the results of these experiments, and Table 4.4 provides the variation about the mean.

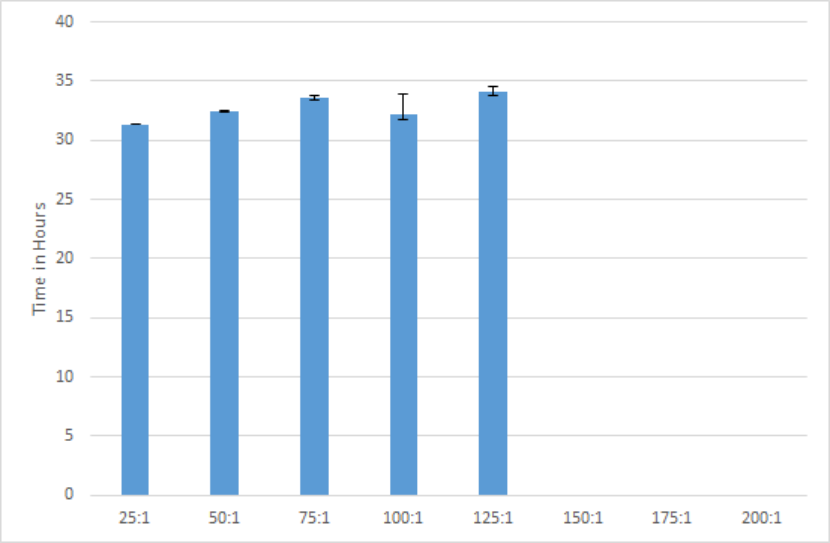


FIGURE 4.15. The average completion times for different agent scalings.

Response Plan Name	**25:1**	**50:1**	**75:1**	**100:1**	**125:1**	**150:1**	**175:1**	**200:1**
rockwall_1	0.00	0.01	0.03	0.62	0.04	-	-	-

TABLE 4.4. The variation about the mean for each of the experiments in 4.15

.

These experiments provided multiple insights into the effects of scaling on completion time. Perhaps the most important effect is that after a certain scaling factor the simulation will not execute. In this case, after scaling to 150:1, an error is thrown by SUMO stating that the route it has chosen does not connect some census block group to its POD. This is due to the way routes are chosen and how the network is scaled. When the network is scaled after it is created via NETCONVERT, if a sufficiently large scaling factor is used, very small road network segments may have their speed reduced to 0 m/s. This would render that segment impassable, as any agent which used that segment would come to a perpetual stop.

However, DUAROUTER uses Djikstra's algorithm for determining a route and prioritizes the lowest travel time. Thus, a road segment that has a speed of 0 m/s could be added to the route, which would then cause the route to be unusable by SUMO, and the simulation is prevented from executing. This was the case for the 150:1, 175:1, and 200:1 scaling factors for Rockwall county.

Another insight into how the scaling factors affect the completion time is shown in Figure 4.15. An increasing trend in the completion time can be seen as the scaling factor increases, with the exception of 100:1. This increasing trend can be potentially explained by the increase in travel time across the road network. Because the speed of vehicles across the network is slowed down to account for the number of individuals an agent represents, it will take an increased amount of simulated time for a single agent to reach their POD and thus the initial time to congestion will increase which will delay the completion time.

Additionally, the different result for 100:1 can potentially be explained by the much higher variation it incurred. However, there could be a variety of factors impacting this particular variation. For instance, the traffic model is sensitive to particular combinations of settings. It was determined in preliminary simulations, that for certain road layouts and for certain scalings, agents would not change lanes correctly which led to significantly delayed travel times. However, despite further investigation it could not be determined what caused this particular variation from the trend. Therefore, the conclusion drawn from these experiments is that using scaling can be sometimes unreliable and unpredictable in terms of effective validation. Due to this, it is not recommended that scaling be used to decrease agent load on the simulations, and this strategy was not used in the final comparison of VERPETS's performance against other analytical methods.

4.3.4. Release Rates

Another factor which potentially affects the early portions of a simulation is the release rate of cars into the simulation. Depending on how often vehicles are released into the simulation could impact how quickly a POD reaches runaway congestion, i.e. the queue perpetually expands until there are no more cars. Thus, experiments were designed to test

the effects of a release rate of one agent every 0, 15, 30, 45, and 60 seconds affects the completion time of the simulation. A 0 release rate indicates that all agents were released at one time. This assumed a 180 second processing time, and used no scaling. Figure 4.16 presents the results.

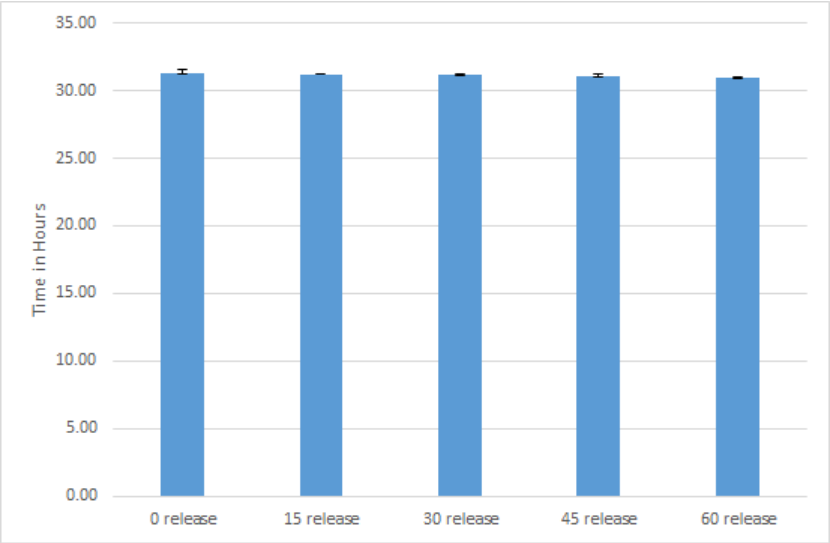


FIGURE 4.16. The average completion times for different agent release rates.

As can be seen from the chart, there was minimal impact on the completion times as a result of the release rates, which indicates that 0, 15, 30, 45, and 60 seconds represent equally safe choices for a release rate in the simulation. This can primarily be attributed to how congestion at the POD is formed. Because there are typically far more census block groups than PODs, as long as the processing rate is consistently below the arrival rate runaway congestion will occur, and the travel time will cease to impact the POD's completion time.

4.3.5. Simulated Time Limit

Due to the dynamics caused by runaway congestion at the PODs, it was inferred that the simulation could be halted early and the completion time extrapolated. If a standard time limit could be identified, then plans could quickly be run and then their completion time estimated. However, because travel time to the POD differs between census block groups, this standard time limit must be experimentally determined.

For this experiment, using SUMO's GUI, a visual examination of each census block

group in a plan was performed. The goal was to determine how long, in simulated seconds, it would take for a POD to reach runaway congestion. It was assumed a POD had reached runaway congestion when all of its booths were fill, and a number of agents equal to the number of booths plus one, had queued up in front of the POD waiting for service. Each plan was simulated with a POD processing time of 180 seconds, and at a 1:1 scaling factor using plans rockwall_1 and denton_1.

Because different vehicle release intervals could affect the time to runaway congestion, three different intervals were used. The first interval test was releasing all agents into the simulation at the same time. This has an effective release rate of one agent every 3 seconds, but may decrease performance as all agents are in the insertion-backlog. The effects of releasing one agent every 30 seconds and one agent every 60 seconds were also examined. Figure 4.17 shows the average time to runaway congestion, and Figure 4.18 shows the max time to runaway congestion.

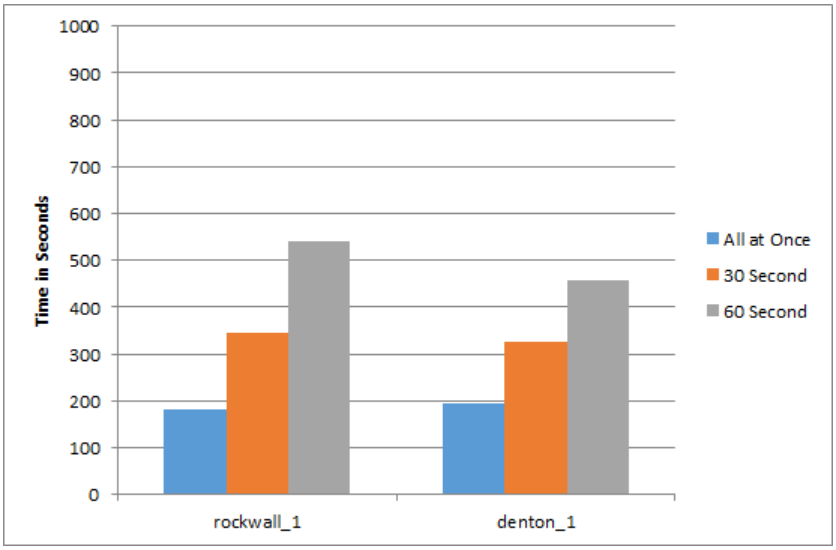


FIGURE 4.17. The average time, in second, to runaway congestion.

Based on the results of those experiments 3600 seconds, or one hour, was chosen as a safe time limit. In addition to determining a time limit, the effects of using that limit were examined. Each of the six plans were simulated with a processing time of 180 seconds, a release rate of 30 seconds, and no scaling. Figure 4.19 shows the results of these simulations

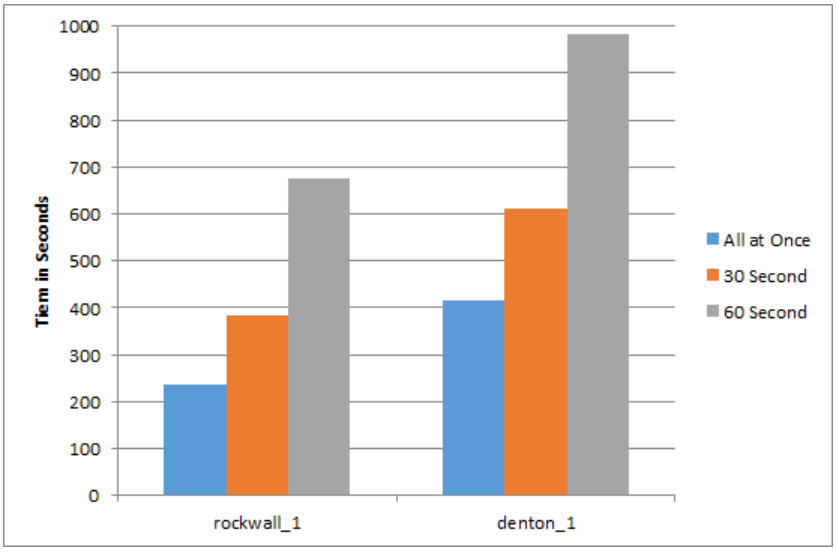


FIGURE 4.18. The max time, in second, to runaway congestion.

and Table 4.5 shows the variation about the mean for each of the experiments.

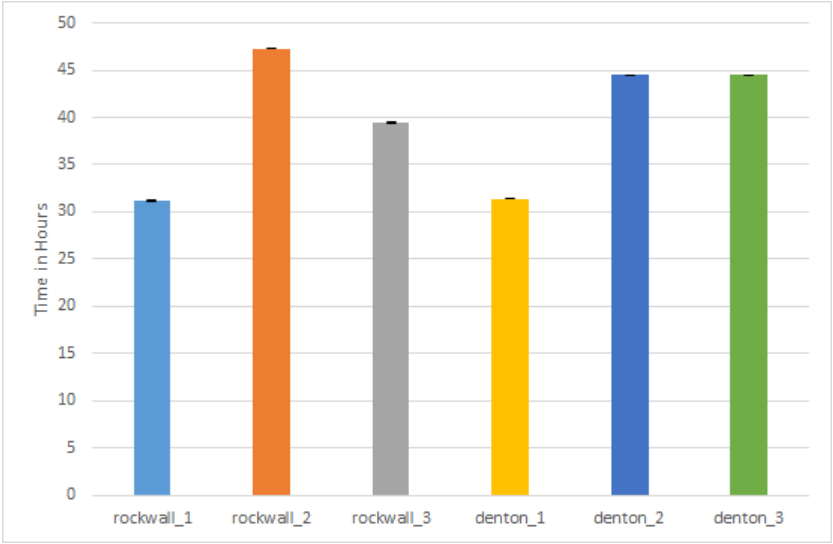


FIGURE 4.19. The average time, in hours, to plan completion using a one hour time limit.

rockwall_1	rockwall_2	rockwall_3	denton_1	denton_2	denton_3
0.00	0.00	0.00	0.00	0.00	0.00

TABLE 4.5. The variation about the mean for each of the experiments in 4.19

.

As the results show, the completion times for each of these experiments was very stable as the variation was almost negligible. Because rounding seemed to have more variability

and more configurations, using a time limit of one hour with a release rate of 30 seconds, was chosen to be the optimal configuration to validate emergency response plans. It will be compared with RE-PLAN's rate based analysis in the next section.

4.3.6. Simulation vs Rate-based Validation

After examining different configurations and methodologies, it was determined that executing a simulation up to a time limit, specifically one hour of simulated time, and then calculating the completion time of the simulation could be an effective method to validate emergency response plans. This conclusion was drawn from results which showed that the time to runaway congestion was consistently less than one-hour. Further, due to the effects of runaway congestion, travel time ceases to affect the completion time of the plan once runaway congestion has occurred. Therefore, a rate-based calculation could be used to determine completion time after the initial period of the simulation where travel time has an impact.

To determine the effectiveness of using a time limit on simulation it must be compared to the rate based calculations that RE-PLAN provides. To perform this comparisson, first, the estimated completion time based on the total number of booths and the total population was calculated according to Equation 4.1, where *Completion Time* is the estimated completion time for the plan, ϕ is the processing time for a single individual, *Total Number of Booths* is the total number of booths across the entire plan, and A is the total population of individuals for the entire plan. This represents a theoretical minimum for how rapidly a plan can be completed if travel time is ignored. Second, for each plan, the booth with the longest completion time, according to RE-PLAN, was determined. This represents the estimated completion time for the response plan. Finally, both of these completion times were compared to the completion time generated by VERPETS which includes travel time. Figure 4.20 shows the results of these experiments.

Based on this comparison, it is apparent that the theoretical completion time is always less than the completion times determined by the other two methodologies. In the case of rockwall_2 the theoretical time was almost seven hours less. This difference between the

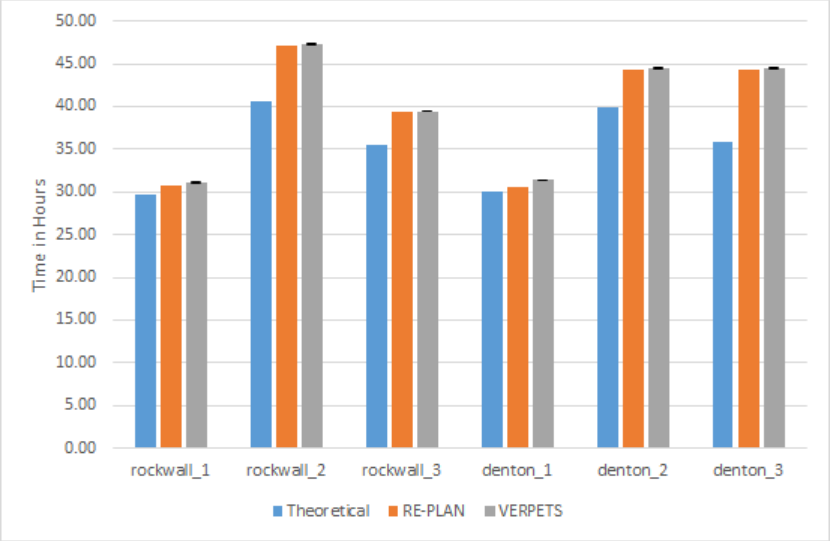


FIGURE 4.20. A comparison between the theoretical, RE-PLAN POD Manager, and VERPETS determined completion times.

theoretical completion time and the other completion times can be attributed to booth distribution amongst the PODs. The theoretical completion time assumes an equal distribution of population assigned to each booth. However, this is not the case as the catchment areas use a closest POD partitioning, which does not ensure an equal distribution of population across PODs.

Finally, when the longest POD time from RE-PLAN is compared with the longest POD completion time from VERPETS there is minimal difference. This is due to the fact that while VERPETS causes PODs to experience a delay before they reach maximum utilization because of travel time, once maximum utilization is reached and runaway congestion begins the main factor determining completion time is the processing rate. Because RE-PLAN only takes this rate into account when determining when a POD will complete the processing of its population, the time it estimates for completion will be very close to one in which travel time is incorporated. Thus, due to the effects of runaway congestion at a POD, both VERPETS and RE-PLAN's estimation of POD completion times, and therefore an emergency response plan completion times, will be similar with VERPETS estimating a slightly longer time than RE-PLAN.

4.4. Response Plan Validation through Scheduling

As Theorem 1 had proved that schedulability implies feasibility, the formation of these schedules could be used to provide a lower bounds to a plan's completion time. In that case, a plan that cannot be scheduled can never be feasible and thus would not need to be simulated. Thus, the methodology presented in Theorem 1 for constructing a schedule has been implemented.

An example of a scheduling for a POD in Rockwall county has been presented in Figure 4.21. In this example, there was a small delay of almost seven minutes before the POD reaches maximum utilization, after which it maintained maximum utilization for 111,357 seconds until no more cars were in the queue to replace a processed agent. The entire population was processed after 111,523 seconds or roughly 30.98 hours. The other PODs in the plan finished after 29.84 hours, 30.11 hours, and 27.31 hours, and thus the plan was theoretically feasible because all PODs completed before the 36 hour time limit.

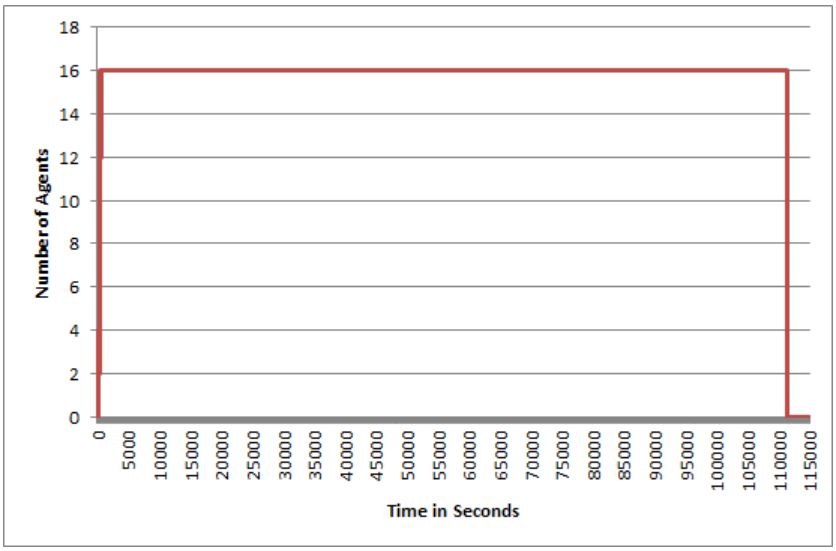


FIGURE 4.21. A chart showing the number of concurrent agents using a POD over time after being scheduled.

Each of the six plans were analyzed using the schedulability method to test for theoretical feasibility. Table 4.6 has presented the results of those experiments. The results generated by the schedulability method were found to align with the those produced by the agent-based simulation, in terms of which plans were or were not feasible. Therefore,

schedulability has been shown to provide an effective lower-bound to a plan's completion time.

Plan	**Feasible/Infeasible**
rockwall_1	Feasible
rockwall_2	Infeasible
rockwall_3	Infeasible
denton_1	Feasible
denton_2	Infeasible
denton_3	Infeasible

TABLE 4.6. A list of whether each plan is theoretically feasible or infeasible according to schedulability.

4.5. Summary of Results

Based on the experiments performed it was shown that agent-based simulations of emergency response plan activation could be effectively performed. First, it was demonstrated that the results of simulating the activation of each individual catchment area was comparable to the results of simulating the activation of an entire plan at once. This allows for a reduction in load on the simulator by allowing for multiple instances of SUMO to be run in parallel. This also improves performance as multiple agents can now be processed in parallel across multiple threads.

Next, two additional strategies for further decreasing agent load were tested: the use of agent scaling and the use of a time limit for simulation. While a methodology for agent scaling was developed and examined, it was determined that the results of simulation using agent scaling were unreliable. Thus, agent scaling was not used further in VERPETS. However, using a time limit for the simulation proved to be effective. It was shown that using a one hour time limit was a safe limit as all examine catchment areas reached runaway congestion before one hour of simulated time. When using the time limit, the results were shown to be stable across all plan activation simulations.

Finally, simulating response plan activation using VERPETS was compared to RE-PLAN's rate based analyses. Both RE-PLAN's analysis and VERPETS's analysis determined a longer plan activation than the theoretical analysis, but this was due to the theoretical

analysis ignoring traffic time and booth distribution across PODs. Between RE-PLAN's analysis and VERPETS's analysis, the results were similar, with VERPETS consistently showing a slightly longer plan activation. Again, this is due to VERPETS not being able to begin processing agents until they arrive at the POD. However, due to the similarity and stability of VERPETSs' results, it can be concluded that VERPETS effectively validated emergency response plans by successfully determining plan feasibility.

Additionally, as was presented in Section 3.1.5, a plan can be determined to be feasible if and only if it is schedulable. Therefore, the methodology from Lemma 1 was implemented and experiments were performed using that implementation. It was determined that this methodology correctly identified feasible and infeasible plans by attempting to schedule the processing of the plan's agents.

CHAPTER 5

SUMMARY AND CONCLUSIONS

Biological emergency response planning plays a critical role in protecting the public from the possible devastating results of sudden disease outbreaks. Thus, the ability to validate that the activation of a plan will successfully provide service to an affected population within some time limit is crucial. While emergency response plan generation softwares such as RE-PLAN [51] or RealOpt [35, 36] do provide some rate based validation analyses, this dissertation sought to develop a agent-based validation system to allow for additional realistic traffic dynamics to be modeled and evaluated. Thus, I developed the VERPETS system to provide effective agent-based simulation and validation of emergency response plan activation.

Towards the goal of generating an agent-based simulation I developed a work flow to convert raw data into a usable format. First, the road network must be downloaded and converted from OSM format into a network file using SUMO's NETCONVERT program. Next, because the road network is a subset of the entire U.S. road network, that potentially allowed for some road segments to not be strongly connected to the rest of the network. Thus, I designed VERPETS to convert the network into a graph, and then a dual graph to effectively use my implementation of Tarjan's strongly connected components algorithm [58]. This removed all weakly connected road segments and prevented the creation of routes that would not connect from a census block group to a POD due to a road segment leading off the road network and not being able to return. A new network file was then created using NETCONVERT to be used by SUMO for routing and simulating. Additionally, I used this network file to derive WKT Linestrings which could be converted to a shapefile for use in a PostgreSQL database. This allowed for the assignment of PODs and census block groups to road segments for routing.

Once the road network was processed, the PODs and census block groups were assigned to their closest road segment via PostGIS queries. This information was then used

by VERPETS to generate trip files which denoted when an agent would leave their census block group, and which POD they were driving to and from. Because trip files are the basis for constructing different style simulations there were several ways in which they could be constructed. One of the early strategies I developed for attempting to decrease the load on the simulator was to divide a plan by its catchment areas and simulate each catchment area in parallel. Experiments showed that there was a negligible difference in plan completion times between a plan that was executed with all agents in one simulation versus a plan what was executed in pieces. Thus, multiple trip files were created for a single plan, one for each catchment area in the plan, and were processed in batches to improve overall performance.

Another method I developed for decreasing load on the simulator was scaling how many individuals of a census block group each agent represents. This scaling affects both the road network and potentially the population of each census block. A plan scaling has two features. The first feature is the scaling ratio of the number of individuals represented by a single agent. Based on this ratio VERPETS alters the road network by scaling the maximum speed on a segment according to the segment's capacity and the scaling ratio. The second feature is rounding which determines how many agents, in the case of a census block group that could not be evenly divided by the scaling ratio, would be generated. Four rounding strategies were examined: always rounding up to the nearest whole agent (ceiling), always rounding down to the nearest whole agent (floor), always rounding to the nearest whole agent (closest);,allowing for partial agents to be created. In this case, rounding had a significant impact on the completion time of the plan, in some cases overestimating it by more than five hours or underestimating it by almost three hours. Thus, the use of partial agents was chosen as a rounding strategy to minimize the effects of changing the represented population of a catchment area.

In determining an appropriate scaling ratio a few challenges were discovered. First, if a plan used a scaling ratio above a certain limit the plan will fail due to some road segments' speeds being scaled to zero meters per second. Second, there was a gradual increase in the completion time as the ratio increased. This could be attributed to the travel time

being increased as the road network speed is scaled down with increased agent scaling. The exception to this trend was the 100:1 scaling which experienced a lower completion time and a higher variance across its mean. It was assumed this was caused by simulation dynamics inside SUMO, but a reliable conclusion could not be determined. This decreased the confidence in using scaling as an effective strategy towards shortening simulation runtime and decreasing load on the simulator. Thus, I concluded that scaling the agents in the simulation would not be a reliable methodology and would not be used further in this research.

Once, the trip files were created, routing could be performed for the agents. SUMO's DUAROUTER tool was used to generate routes along the road network using Djikstra's algorithm and using travel time as a weight. The route file was then edited by VERPETS to include POD locations and processing times to each agent in the file. VERPETS then created additional files to describe POD locations and the number of booths at a POD used for concurrent processing. With these files generated an early analysis of the emergency response plan could be performed by VERPETS. Based on the population of each census block group and the route a census block group took to its POD, the load on each segment of the road network could be calculated. This provided the information to generate maps, as seen in Figure 5.1, which allow emergency managers and planners to determine where heavily used sections of the network would be for a given plan. This could then be used to direct additional resources, such as police or tow trucks, before a plan was activated to prevent potentially costly, repeated road blockages.

Once the network file, route file, and additional file were created, the SUMO configuration file could be generated which provided the necessary information to the simulator to execute a simulation. Because some plans may contain many catchment areas, a plan's activation can be simulated in batches of catchment areas so as to not overwhelm a computer with too many concurrent simulations. Because a plan is considered completed once all agents of the population have been processed at a booth, additional processes are forked by VERPETS to monitor the output of each simulation and allow for the simulation to be halted before all agents have left the simulation.

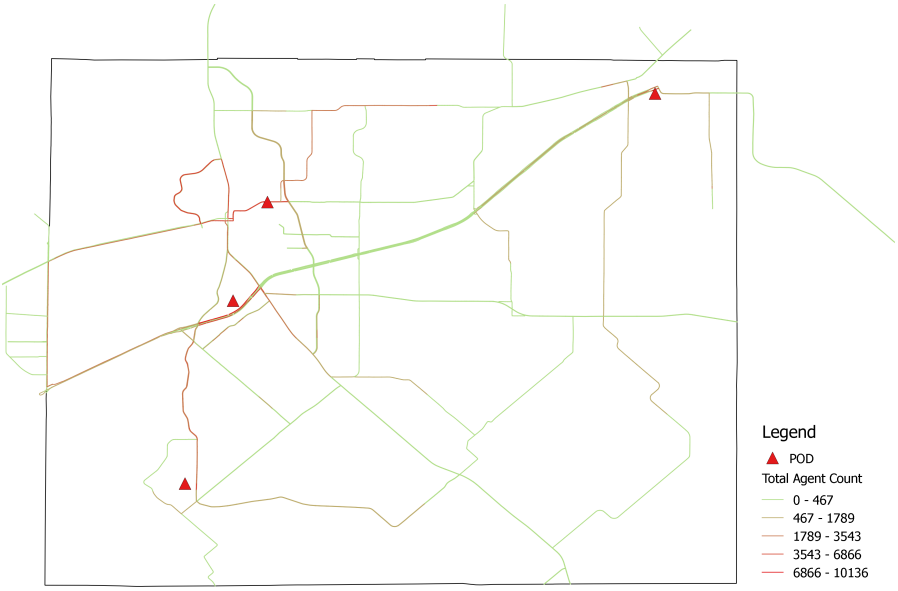


FIGURE 5.1. The load on the road network for Rockwall county during plan activation.

As an alternative to scaling, I developed another strategy to decrease the runtime of a simulation. I noticed early on that one of the dynamics encountered when performed agent based simulations was that of runaway congestion. Thus, because a POD experiences perpetual congestion after some time this causes the POD to experience continuous maximum utilization until almost all agents are processed. Therefore, a simulation can be halted early and a rate based calculation can be performed to estimate the completion time of the plan. Based on experimental data, I determined one hour would be a safe time list.

Using a halting time of one hour, a processing time of 180 second, and no scaling, the average completion times generated by VERPETS for six plans were compared against the completion times generated by RE-PLAN and by Equation 4.1. It was determined that Equation 4.1 provided a theoretical minimum completion time for which a plan could not perform better than, but this ignored travel time and population and booth distribution. RE-PLAN's estimated completion time was consistently slightly lower than VERPETS's, but the difference can be explained by the travel time delay before a POD reaches runaway congestion. Thus, due to the effects of runaway congestion at a POD, both VERPETS and

RE-PLAN's estimation of POD completion times, and therefore an emergency response plan completion times, will be very similar with VERPETS estimating a slightly longer time than RE-PLAN.

In addition to showing how VERPETS's agent-based simulations compared to RE-PLAN's rate based calculations of emergency response plan completion time, I discovered that a third methodology could be employed. Using Theorem 1, I proved that a plan's feasibility was directly impacted by its schedulability. This theorem described a methodology for attempting to schedule each agent in a plan at a POD for processing and took into account travel time and processing time. If a plan was deemed schedulable, then it was theoretically feasible, and if it was unschedulable then it was theoretically infeasible. This methodology was implemented, and its prediction of plan feasibility aligned perfectly with both RE-PLAN's and VERPETS determination of a plan's feasibility.

5.1. Contributions

My dissertation has shown both how agent-based simulations can be used to validate emergency response plans and how agent-based simulations of emergency response plan activations compare with analytical examinations. I have created the system Validating Emergency Response Plan Execution Through Simulation (VERPETS) to provide a computational solution for taking raw road network data, population distributions, and response plan data and performing simulations of emergency response plan activation. This system currently allows for the examination of plans assuming a constant rate of release from the census block groups. However, the system could be configured to allow for the examination of plans using staggered release rates to provide emergency manager and planners with insight into how staggering population release can affect plan outcomes. Further, the system could be configured to allow for random release rates, and potentially allow for agents to disobey instructions and travel to PODs other than the one they were assigned to. This can help determine the effects of individuals failing to follow the plan, and analyze how tolerant a plan is to these types of problems.

Further, VERPETS can be reconfigured to analyze the effects of portions of the road

network being disabled. As this was originally part of the design of VERPETS, re-enabling this functionality would be straightforward, though additional research would need to be performed in order to determine an effective methodology, as will be discussed in the Future Work section. This road network analysis could help emergency manager and planners better analyze the robustness of the road network, and determine where critical services such as police or tow trucks could be positioned to have the greatest impact if blockages occurred.

In addition to the VERPETS system as a whole being created, certain specific functions can be used independent of the system to aid developers and researchers. For example, my novel strategy of renaming OSM ways to include unique ID numbers allows for the comparison of results involving the road network across different OSM processing tools. I developed this into a PERL script and hosted it on a public Git repository to allow other researchers using OSM data to utilize this tool. Additionally, my methodology for removing isolated and non-returning road segments by generating the dual graph of the network and performing Tarjan's strongly connected components algorithm could be converted into a PERL script. This would allow researchers and developers to include this functionality in their systems in order to solve the problem of the introduction of isolated edges caused by only using portions of a road network.

5.2. Limitations

The limitations of the methodologies and systems described in this dissertation stem from multiple sources surrounding plan activation simulation. First, despite having accurate road network data via OSM, some finer details were not included in that dataset. Specifically, not all mapped road segments contained speed limit information. Due to this, NETCONVERT infers the speeds for each road segment based on the type of road. However, this speed does not always accurately reflect the changes in the posted speed limit that may occur along named roads. For example, I-35 alternates between 70, 65, 60, and 55 mph in Denton County, but NETCONVERT would apply the same speed to all of those segments when generating the network because it does not know what the actual speed is. While the assignments of road segment speed SUMO makes are at least consistent, having additionally

finer grain road network speed information would be beneficial to more accurately reflect real world travel times. Thus, currently, it can be assumed that the travel time estimates may be higher or lower than they actually should be.

The simulator chosen for this research also posed some challenges. Despite some of the tools that SUMO provides being potentially multi-threaded, the simulator itself appears to be single-threaded which causes it to suffer significant slow-downs when simulating large numbers of agents concurrently. This necessitates either the use of scaling or parallelization to decrease the load on the simulator. Additionally, while SUMO does offer the ability to dynamically route cars during simulation, this functionality causes considerable delays in the simulation as it needs to repeatedly route each car individually. Thus, dynamic routing was not used for this dissertation due to this situation. Consequently, traffic routing was not as realistic as it could be, which could lead to current travel time estimates being higher than they should be. However, even when using dynamic routing simulations still encounter runaway congestion, which negates the impact of travel time after a certain period of time.

Additionally, both Theorem 1 and the methodology in 3.1.6 assume a constant travel time for each agent as it travels along the network. While an average travel time can be estimated for each route, it may not accurately reflect fluctuations in agent speed due to the dynamic nature of traffic. Another limitation of using the release rate calculation methodology is that perfect assignment of booths to block groups is a $\binom{n}{m}$ operation and hence too expensive. A heuristic can be used for this assignment but it may not yield the absolute minimum waiting time at the booths though it will ensure that the simulation is congestion free.

Further, the dynamic of runaway congestion had a significant impact on the design and outcome of this research. The fact that travel time ceases to impact POD utilization and completion time after some time period allows for early calculation of the completion time if an early halting strategy is employed. However, because of runaway congestion and the fact that agents will not leave the queue once they enter it, some agents may wait in the queue for 20+ hours. Yet, it can still be argued that significant congestion would occur

at a POD during an actual response plan activation, which would lead to maximum POD utilization. Even if agents arrived at a congested POD, waited for some time, went home, and returned later, it is still likely that persistent congestion would exist at the POD in sufficient quantity to cause the POD to be at maximum efficiency for a prolonged duration.

Finally, as the VERPETS system is constructed on multiple models and with certain assumptions, this presents some limitations in the use of VERPETS to validate emergency response plans outside of those described. For example, this research assumed that all plans were created using a closest POD partitioning which reduces overall travel time. However, some plans may be generated to optimize other metrics, such as ensuring an equal population distribution across all PODs or optimizing for reachability and walkability. In this case, analysis would need to be performed on road usage overlap before assuming the activation of each catchment area of the plan could be simulated in parallel with minimal impact from neighboring catchment areas. Also, selecting SUMO as the simulation engine presented restrictions on how agents and routes could be modeled for plan activation, and thus how VERPETS processes and interprets these data. Therefore, the ability and suitability for VERPETS to validate biological emergency response plans of a variety of styles and configurations implies that the system would need to be reconfigured and, in some cases, extended in future research. Some of these proposed extensions and reconfigurations are presented in Section 5.3.

5.3. Future Work

Based on the findings, the development of VERPETS and as a result of the experiments performed, there are further opportunities for research. First, one of the original aims of this research was to be able to identify problematic road segments and combinations within a response plan automatically. This was later determined to be a combinatorial issue due to the large number of road segments which could be affected, and due to the time it will take to test all of those situations. This is further compounded by the time required to regenerate the road network for each simulation. It was proposed that using methodologies from software testing, such as code coverage, could be applied. In this case, statement coverage

strategies [43, 7] could be used to attempt to test all road segments after a sufficient number of simulations were performed. This potentially has the benefit of requiring fewer simulations and fewer regenerations of the road network than testing all possible combinations of road segments. Additionally, this would allow a machine learning model to be constructed to direct the experimentation based on simulation results. Active learning [55, 11, 12] would be an effective methodology given its ability to learn from limited training data and request new experiments to improve its confidence. VERPETS was originally designed and constructed to work in conjunction with an active learning agent, so it is currently possible to allow for this type of testing.

Following the original design, several randomly generated simulations would be executed and those results recorded to produce an initial training set of data to train the learning agent. Next, the learning agent would attempt to classify portions of the road network, and those portions it was least confident in classifying would be marked for experimentation. Because VERPETS is currently designed to produce simulation results without any user input after the configuration file is read in, the ability to automatically generate new simulations, and thus results, is already available. The learning agent could then be allowed to cycle between classifying the network and directing experiments until a sufficient level of confidence in its classification were reached, or until a user defined number of cycles were performed.

In addition to code coverage and active learning, the elimination of runaway congestion could lead to additional insights into POD and road network performance. The methodology outlined in 3.1.6 provides a strategy for minimizing congestion at the POD by controlling the release rates from the census block groups. By implementing this methodology the effects on the road network caused by different impediments could be more thoroughly examined, as runaway congestion would no longer reduce the relevance of travel time. Coupled with more accurate speed limit data, this could bring additional realism to the model and reduce the lag in the simulation brought on by too many agents being simulated at the same time.

Finally, if RE-PLAN were expanded to plan for additional types of emergencies, such as evacuations before and during disasters or the distribution of other types of supplies after disasters, VERPETS could be similarly expanded to simulate these new plans. In the case of hurricanes, planning for contraflow lanes and evacuation routes must be planned and constructed before a hurricane arrives [10]. This necessitates evaluation of these routes and plans in advance so that upon activation, potential problems have already been identified and mitigated. Additionally, in the wake of the hurricane, supplies such as fresh water, and medical supplies must be distributed in a timely and effective manner. Potentially these supplies can be distributed via mobile distribution centers, and how to best serve portions of the population quickly could be tested via simulation. Thus, VERPETS could be restructured to both test evacuation strategies and mobile distribution of supplies strategies.

APPENDIX

POD STANDARDS

1	**NUMBER AND LOCATION OF PODS**
1.1	"The jurisdiction shall estimate the number of people who will likely come to PODs to pick up medication, along with their geographic distribution."
1.2	Number of PODs must be greater than or equal to the number of persons needing treatment at a POD divided by POD throughput multiplied by 24 hours.
1.3	"All POD locations shall meet relevant SNS site guidelines and security criteria."
2	**INTERNAL POD OPERATIONS**
2.1	"Jurisdictions must have at least one viable and exercised rapid [dispensing] protocol that addresses the following minimal functions: directing clients through the POD; deciding which medication to dispense; dispensing medication; and disseminating information about the medication."
2.2	"Jurisdictions shall ensure that legal and liability barriers to rapid dispensing are identified, assessed, prioritized, and communicated to those with the authority to address such issues."
2.3	"Jurisdictions shall ensure they have viable and exercised procedures for selecting an appropriate dispensing protocol."
3	**POD STAFFING**
3.1	"Jurisdictions shall estimate the number of individuals who are likely to visit each POD location and determine the required hourly throughput at each POD."
3.2	"Using a combination of exercises and/or computer models, jurisdictions shall determine and verify the number of staff required to administer prophylaxis to the population identified pursuant to Standard 1.1."
3.3	"Jurisdictions shall recruit sufficient command staff, and provide plans for recruiting and training of spontaneous unaffiliated volunteers, in sufficient numbers to operate all the planned PODs in the jurisdiction at the levels of throughput required to meet the CRI timeline."

3.4	"Jurisdictions shall assess the availability of the command staff on their call-down rosters on a quarterly basis, via a no-notice call-down drill."
4	**POD SECURITY**
4.1	"Site security assessments shall be conducted on every POD location in coordination with the agency(ies) responsible for security functions at the PODs."
4.2	"The agency(ies) responsible for security functions at PODs shall be consulted on the security aspects of the overall mass prophylaxis plan."
4.3	"Law enforcement in the form of sworn uniformed officers shall maintain a physical presence at each POD location."

TABLE A.1. A list of POD standards developed to improve POD performance, effectiveness, and safety [24].

REFERENCES

[1] *Project BioShield Act*, 2004.

[2] Pam Adams, *Mass flu vaccination drill also is practice for an emergency.*

[3] Alfred V. Aho, Jeffrey D. Ullman, and John E. Hopcroft, *Data Structures and Algorithms*, 1rst ed., Addison-Wesley, Reading, MA, 1983.

[4] American Red Cross, *Red Cross Responds to Help People after Floods and Tornadoes*, 2014.

[5] ARGONNE NATIONAL LABORATORY, *The Repast Suite*, 2015.

[6] Vidhya Balasubramanian, Daniel Massaguer, Sharad Mehrotra, and Nalini Venkatasubramanian, *DrillSim: a simulation framework for emergency response drills.*, Springer Berlin Heidelberg, 2006.

[7] B. Beizer, *Software testing techniques*, Dreamtech, 2003.

[8] Richard Burr, *S.3678 - Pandemic and All-Hazards Preparedness Act*, 2006.

[9] Centers for Disease Control and Prevention, *Division of global migration and quarantine (dgmq)*, Dec 2016.

[10] ———, *Notes on the Interim U.S. Guidance for Monitoring and Movement of Persons with Potential Ebola Virus Exposure*, 2016.

[11] David Cohn, Les Atlas, and Richard Ladner, *Improving generalization with active learning*, Machine Learning 15 (1994), no. 2, 201–221.

[12] David a Cohn, Z Ghahramani, and Michael I Jordan, *Active Learning with Statistical Models*, Journal of Artificial Intelligence Research 4 (1996), 129–145.

[13] OpenStreetMap contributors, *OpenStreetMap*, 2017.

[14] datapolitan, *Census, sample, survey, tract, block, zip code a primer*, Aug 2013.

[15] Urška Demšar, *Centrality measures and vulnerability of spatial networks*, Iscram (2007), 201–209.

[16] E. W. Dijkstra, *A note on two problems in connexion with graphs*, Numerische Mathematik 1 (1959), no. 1, 269–271.

[17] Krzysztof Drewniak, Joseph Helsing, and Armin R. Mikler, *A Method for Reducing the Severity of Epidemics by Allocating Vaccines According to Centrality*, arXiv:1407.7288 [physics] (2014), 341–350.

[18] EdrawSoft, *Hotel fire escape plan.*

[19] Emergency Management Institute, *The Four Phases of Emergency Management*, Tech. report, 2015.

[20] Environmental Tectonics Corporation, *Advanced Disaster Management Simulator*, 2016.

[21] Federal Bureau of Investigation, *Amerithrax or Anthrax Investigation.*

[22] FEMA, *Building codes.*

[23] Center for Disease Control and Prevention, *Strategic national stockpile*, Mar 2017.

[24] Centers for Disease Control and Prevention, *Point Of Dispensing (POD) Standards*, Tech. Report April, Centers for Disease Control and Prevention, 2008.

[25] ———, *Quarantine and isolation*, Mar 2014.

[26] ———, *Centers for Disease Control and Prevention (CDC) BP4 Medical Countermeasure (MCM) Operational Readiness Review (ORR) Guidance*, Tech. report, Kansas Department of Health and Human Services, 2016.

[27] Center for the Advancement of Distance Education, *The pod game.*

[28] Brian Gardener, *Transportation Analysis and Simulation*, 2006.

[29] Robert Hilbrich, *SUMO Simulation of Urban MObility.*

[30] Andreas Horni, David Charypar, and Kay.W. Axhausen, *Variability in Transport Microsimulations InvestigatedWith the Multi-Agent Transport Simulation MATSim*, 2013.

[31] Sanjay Jain, *An Integrating Framework for Modeling and Simulation for Emergency Response*, Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693) (2003), no. April, 1068–1076.

[32] Sanjay Jain and Charles McLean, *A framework for modeling and simulation for emergency response*, Proceedings of the 35th conference on Winter simulation: driving innovation, 2003, pp. 1068–1076.

[33] GEORGE F. JENKS and FRED C. CASPALL, *Error on choroplethic maps: Definition,*

measurement, reduction, Annals of the Association of American Geographers 61 (1971), no. 2, 217–244.

[34] Daniel Krajzewicz and G Hertkorn, *SUMO (Simulation of Urban MObility) An open-source traffic simulation*, ... Symposium on Simulation ..., 2002, pp. 63–68.

[35] Eva Lee, *Overview of realopt software enterprise*.

[36] Eva K Lee, Siddhartha Maheshwary, Jacquelyn Mason, and William Glisson, *Large-Scale Dispensing for Emergency Response to Bioterrorism and Infectious-Disease Outbreak*, Interfaces 36 (2006), no. 6, 591–607.

[37] Michal Maciejewski and Kai Nagel, *Towards Multi-Agent Simulation of the Dynamic Vehicle Routing Problem in MATSim*, Parallel Processing and Applied Mathematics 7204 (2012), 551–560.

[38] ———, *Simulation and dynamic optimization of taxi services in MATSim*, TRANSPORTATION SCIENCE 13 (2013), no. 0.

[39] Christopher Nelson, Edward W. Chan, Anita Chandra, Paul Sorensen, Henry H. Willis, Kathering Comanor, Hayoung Park, Karen A. Ricci, Leah B. Caldarone, Molly Shea, John A. Zambrano, and Lydia Hansell, *Recommended Infrastructure Standards for Mass Antibiotic Dispensing*, Tech. report, 2008.

[40] Pillemer F. Nelson C, Parker A, Shelton S, Chan E, *Analysis of the Cities Readiness Initiative TR-1200*, RAND Corporation, 2012.

[41] New York City Health Department, *RAMPEx Exercise in New York City : What You Need to Know*, Tech. Report August, New York City Health Department, New York City, 2014.

[42] Pat Nonnenmacher, *Local Technical Assistance Review Summary*, Tech. report, Centers for Disease Control and Prevention, Yolo County, 2010.

[43] S. C. Ntafos, *A comparison of some structural testing strategies*, IEEE Transactions on Software Engineering 14 (1988), no. 6, 868–874.

[44] Department of Defence, *Dod modeling and simulation (m&s) verification, validation, and accreditation (vv&a)*, DODI, 1996.

[45] U.S. Department of Health & Human Services, *Medical countermeasures*, Oct 2016.

[46] The Governor's Office of Homeland Security and Emergency Preparedness, *Family disaster evacuation guide*.

[47] The City of New Orleans, *Plan for emergencies*.

[48] United States. Bureau of the Census, *Geographic areas reference manual*, U.S. Dept. of Commerce, Economics and Statistics Administration, Bureau of the Census, 1994.

[49] Office of Public Health, *Free Flu Shots Available One Day Only*, 2007.

[50] Office of Public Health Preparedness and Response, *Cities Readiness Initiative*, 2015.

[51] Martin O'Neill II, Armin R Mikler, and Tamara Schneider, *Re-Plan - An Extensible Software Architecture to Facilitate Disaster Response Planning*, Traffic 44 (2004), no. 12, 1569–1583.

[52] Open Source Initiative, *The Open Source Definition*, 2007.

[53] OpenStreetMap, *List of OSM-based services*, 2017.

[54] Kevin Quinn, *HOUSTON HOSPITALS PRACTICE DRILLS FOR EBOLA*, 2014.

[55] Burr Settles, *Active Learning Literature Survey*, Machine Learning 15 (2010), no. 2, 201–221.

[56] M. Sharir, *A strong-connectivity algorithm and its applications in data flow analysis*, Computers & Mathematics with Applications 7 (1981), no. 1, 67–72.

[57] Boris Shulgin, Lewi Stone, and Zvia Agur, *Pulse vaccination strategy in the sir epidemic model*, Bulletin of Mathematical Biology 60 (1998), no. 6, 1123–1148.

[58] R. E. Tarjan, *Depth First Search and Linear Graph Algorithms*, jun 1972, pp. 146–160.

[59] Eugenia Tognotti, *Lessons from the History of Quarantine, from Plague to Influenza A*, Emerging Infectious Diseases 19 (2013), no. 2, 254–259.

[60] United States Census Bureau, *Annual Estimates of the Resident Population for Counties: April 1, 2010 to July 1, 2014*, Tech. report, 2015.

[61] SUNY Upstate Medical University, *Four phases of emergency management*, Feb 2016.

[62] U.S. Department of Health & Human Services, *Strategic National Stockpile (SNS)*, 2016.

[63] Aditya Vaidya, Angel D. Bravo-Salgado, and Armin R. Mikler, *Modeling climate-*

dependent population dynamics of mosquitoes to guide public health policies, Proceedings of the 5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics (New York, NY, USA), BCB '14, ACM, 2014, pp. 380–389.

[64] Jean Weinberg and Levi Fishman, *Health Department Conducts the Largest No-Notice Emergency Response Exercise in New York City History*, aug 2014.

[65] Brian Wolshon, Transportation Resiliency, Joseph Lefate, Hana Naghawi, Thomas Montz, and Vinayak Dixit, *Application of TRANSIMS for the Multimodal Microscale Simulation of the New Orleans Emergency Evacuation Plan*, Tech. report, New Orleans, 2009.

[66] Yolo County Emergency Response, *MOCK Avian Influenza/Vaccine Fact Sheet*, Tech. report, Yolo County Emergency Response, Davis, 2004.