Real-Time Transit Vehicle Routing Optimization in Intermodal Emergency Evacuations

by
Li Zhang, Ph.D., P.E.
Associate Professor

Department of Civil and Environmental Engineering
P.O. Box 9546, Mississippi State University
220 Walker Engineering Building
Mississippi State, MS 39762
Phone: (662) 325-9838
Fax: (662) 325-7189
Email: lzhang@cee.msstate.edu

NCITEC Project No. 2012-15

June 2016

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Abstract

Since Hurricane Katrina, extensive studies have been conducted aiming to optimize transit vehicle routing in an emergency evacuation. However, the vast majority of the studies focus on solving the deterministic vehicle routing problem, in which all evacuation data are known in advance. These studies are generally not practical in dealing with real-world problems that involve considerable uncertainty in the evacuation data set. In this project, a SmartEvac system is developed for dynamic vehicle routing optimization in an emergency evacuation. The SmartEvac system is capable of processing dynamic evacuation data—such as random pickup requests, travel time change, and network interruptions—in real time. The objective is to minimize the total travel time for all transit vehicles.

A column generation based online optimization model is integrated into the SmartEvac system. The optimization model is based on two structures: a master problem model and a sub-problem model. The master problem model is used for route selection from a restricted routes set, while the sub-problem model is developed to progressively add new routes into the restricted routes set. The sub-problem is formulated as a shortest path problem with capacity constraints and is solved using a cycle elimination algorithm. When the evacuation data are updated, the SmartEvac system will reformulate the optimization model and generate new routes set based on the existing routes set. The computational results on benchmark problems are compared to the results from other studies in the literature. The SmartEvac system outperforms the other approaches on most of the benchmark problems in terms of computation time and solution quality.

CORSIM simulation is used as a test bed for the SmartEvac system. CORSIM Run-Time-Extension is developed for communications between the simulation and the SmartEvac system. A case study of the Hurricane Gustav emergency evacuation is conducted, where different scenarios corresponding to the different situations that happened in the Hurricane Gustav emergency evacuation are proposed to evaluate the performance of the SmartEvac system in response to real-time data. The average processing time is 28.9 seconds, and the maximum processing time is 171 seconds, which demonstrates the SmartEvac system's capability of real-time vehicle routing optimization on an Intel Core I5 Laptop. The dynamic vehicle routing optimization model is deployed and implemented to a web-based online service system to allow transit agencies or drivers to exchange necessary data.

Table of Contents

INTRODUCTION 1

LITERATURE REVIEW 4

I. Evacuation Modeling 4

Macroscopic Simulation Model 4
Microscopic Simulation Model 6
Mesoscopic Simulation Model 7

II. Vehicle Routing Problem 9

Exact Algorithms 9
Heuristics 12
Dynamic Vehicle Routing 12

III. Web-based Transit Service 14

IV. Transit in Intermodal Transportation 15

METHODOLOGY 17

V. Problem Statement 17

VI. Model Development 17

Base Model 18
Column Generation Model 23
CDVRPPD Model 31
2-Stage Intermodal Evacuation Model 39

VII. Solution Algorithm 40

Initialization 42
2-Cycle Elimination 45
Computational Results 51

SIMULATION AND CASE STUDY 54

VIII. CORSIM Network Development 54

IX. CORSIM Simulation Development 55

X. Results of Case Study 58

Scenario 1 58

Scenario 2 60
Scenario 3 66

XI. Results Analysis 77

Computation Time 77
Response to Evacuation Information Updates 78
Comparison with Real Evacuation Results 79

WEB-BASED INTERFACE DEVELOPMENT 81

XII. The web-based interface design 81

XIII. System Implementation 82

WCF API 82
WCF Host 83
WCF Clients 84

CONCLUSION AND RECOMMENDATIONS 88

REFERENCES 90

List of Figures

Figure １Online Transit Planning System Architecture 15

Figure ２A Simple Representation of Vehicle Routes 20

Figure ３Columns Set Augmentation..29

Figure ４Time Intervals over the Planning Horizon 32

Figure ５Column Generation Approach Flow Chart...41

Figure ６Insertion Algorithm Flow Chart 44

Figure ７Illustration of Dominance Rule 47

Figure ８Distribution of Shelters and Pickup Points with Registered Evacuees 54

Figure ９Data Used in the CORSIM Simulation 56

Figure １０ CORSIM Network of Gulfport Region 57

Figure １１Travel Time Comparison between Simulation, Google Map, and Historical Data..58

Figure １２ Results from the Scenario 1...59

Figure １３ Transit Vehicle Routes after Re-optimization in Scenario 2 t1.....................60

Figure １４Comparison of Route 1 between Scenario 1 and Scenario 2 t1 61

Figure １５ Comparison of Route 3 between Scenario 1 and Scenario 2 t1 61

Figure １６ Comparison of Route 5 between Scenario 1 and Scenario 2 t1 62

Figure １７ The Spatial Distribution of Zones of Pickup Points....................................62

Figure １８ CORSIM Network with Unregistered Evacuees..66

Figure １９ CORSIM Network of Scenario 3(a) 67

Figure ２０Transit Vehicle Routes after Re-optimization in Scenario 3(a) Interval t1..68

Figure ２１ Comparison of Route 1 between Scenario 2 t1 and Scenario 3(a) t1............69

Figure ２２ Comparison of Route 4 between Scenario 2 t1 and Scenario 3(a) t1 69

Figure ２３ Comparison of Route 5 between Scenario 2 t1 and Scenario 3(a) t1 69

Figure ２４ CORSIM Network of Scenario 3(b) 72

Figure ２５ Updated Transit Routes in Scenario 3(b) t3 73

Figure ２６ Comparison of Route 1 between Scenario 3(a) t3 and Scenario 3(b) t3 74

Figure ２７ Comparison of Route 2 between Scenario 3(a) t3 and Scenario 3(b) t3 74

Figure ２８ Comparison of Route 3 between Scenario 3(a) t3 and Scenario 3(b) t3 74

Figure ２９ Comparison of Route 5 between Scenario 3(a) t3 and Scenario 3(b) t3 75

Figure ３０ Computation Time in Scenario 2, Scenario 3(a), and Scenario 3(b) 77

Figure ３１ Response Time in Scenario 2 with Fixed Interval and Dynamic Interval 79

Figure ３２ System environment of web-based interface 81

Figure ３３ Structural Diagram 82

Figure ３４ Screen Shot of WCF Host Application 84

Figure ３５ Website client is ready for connecting server host 85

Figure ３６ Website client get optimization from server host 86

Figure ３７ Windows form client get optimization from server host 87

List of Tables

Table 1 Computational Results 51

Table 2 Results Comparison of Agarwal, Mathur, and Salkin (AMS) and CE 52

Table 3 Results Comparison of Bixby and CE 52

Table 4 Results Comparison of Hadjiconstantinou, Christofides, and Mingozzi (HCM) and CE 52

Table 6 Results of Scenario 2 with Fixed Interval..63

Table 7 Figure 10 Results of Scenario 2 with Dynamic Interval.......................................65

Table 8 Results of Scenario 3(a)...70

Table 9 Figure 12 Results of Scenario 3(b) 75

Table 10 Comparison of SmartEvac system’s results and CTA’s record..........................80

Real-Time Transit Vehicle Routing Optimization in Intermodal Emergency Evacuations

INTRODUCTION

In 2005, transit could have played an important role by assisting in the evacuation of an estimated 150 to 200 vulnerable residents in the Gulf Coast region who lacked access to a private vehicle during Hurricane Katrina. In response to the lessons learned from Hurricane Katrina, transit agencies are taking more active procedures in the evacuation of transit-dependent populations.

The Federal Transit Administration has established the Advanced Public Transportation Systems (APTS) program to encourage development and implementation of innovative technologies and strategies to improve transit service. Transit agencies across the nation—including APTS in Los Angeles, the Regional Transportation District (RTD) in Denver, the Milwaukee County Department of Public Works Transportation Division (MCTD), the Kansas City Area Transportation Authority (KCTA), the Maryland's Mass Transit Administration (MTA), and the Dallas Area Rapid Transit (DART)—are implementing the Automatic Vehicle Location (AVL) system to monitor both the location and performance of transit vehicles.

The AVL system could be further develop for transit emergency evacuation plans; register transit-dependent populations; identify the maximum number of transit-dependent populations; consider school buses and drivers for meeting the surge demands of emergency evacuation; develop a plan especially for evacuating people with special needs (e.g., the disabled, the elderly); develop standby emergency service contracts to fill remaining transit service gaps; build real time communication between transit drivers and emergency managers, as well as the public; and coordinate with state and local departments of transportation to provide dedicated lanes to facilitate transit trips in an emergency evacuation (White, 2008).

In a US DOT study to evaluate the evacuation plans in the Gulf Coast region, many factors limit the role of transit. Firstly, the damage to the transportation system, which includes major bridge damage, would seriously impede the progress of the transit evacuation. Another limit comes from the congestions at peak periods during the workday. The last limit would be the unpredictability of evacuation data. Even for a hurricane emergency evacuation with advance notice, when the hurricane will make landfall and what its path will be remain uncertain in the planning stage. The number of evacuees may be very significantly dependent on the size and severity of the hurricane as well.

Taking these factors into consideration, an approach to consider for transit operations in emergency scenarios is the Capacitated Dynamic Vehicle Routing Problem with Pickup and Delivery (CDVRPPD), which is an extension of the Vehicle Routing Problem (VRP). The CDVRPPD involves solving the vehicle routing problem with pickup and delivery in a real-time environment. The SmartEvac system is developed based on CDVRPPD. It takes real-time evacuation data available to the transit agencies and uses this data to output vehicle routes and pickup locations in an emergency evacuation. The SmartEvac system contains several real-time routing features, including real-time evacuation data collection and processing, demand-responsive transit vehicle routing and scheduling, and real-time response to transportation network interruptions.

The SmartEvac is a real-time transit evaluation system that can be adopted for emergency evacuations of mid-size cities. The system is designed to support an effective delivery of transit service. The SmartEvac system focuses on optimizing fleet planning, scheduling, and operations. Approved transit agencies will be able to access the supporting tools at any time, as long as they are connected to the Internet. Real time traffic information and evacuee information can be used and updated to generate the routing plan. In order to improve transit route running times, a CDVRPPD model is implemented in the SmartEvac system to optimize the total travel time. The CDVRPPD model is based on a master problem–sub-problem structure. The master problem is formulated as a Set Covering (SC) model that is used for routes selection from a restricted routes set. The sub-problem is formulated as an Elementary Shortest Path Problem with Capacity Constraint (ESPPCC) model that progressively adds new routes into the restricted routes set. The SmartEvac system is validated through a case study of the Hurricane Gustav evacuation procedures in Gulfport, MS. A CORSIM simulation is conducted as a proof-of-concept to demonstrate the SmartEvac system's feasibility in a dynamic environment. CORSIM Run-Time-Extension (RTE) is developed as a communication interface that enables a data exchange between CORSIM simulation and the SmartEvac system. Different scenarios corresponding to the different situations that happened in the Hurricane Gustav emergency evacuation are proposed to evaluate the performance of the SmartEvac system in response to real-time data. Furthermore, the SmartEvac system could increase operating efficiency, service reliability, and resilience of transit service in an emergency; and improve response to surge demands and service disruptions. Overall, the SmartEvac system would improve the efficiency and safety of the transit service, which would lead to a successful emergency evacuation.

To consider the real world actual requirements, the SmartEvac system would provide remote functions to web-based service. The transit agencies have access to upload traffic input data or download an optimized routing plan from a website/web-based application. The optimized routing plan (once approved by the manager) will be sent to the transit driver's smart phones with Google Navigation. First, the web service will be designated to support the transit operations in small cities in Mississippi. After a period of test runs and updates in the future, the service could be introduced to a wide range of small to medium cities throughout the United States. To achieve the web-based design, the SmartEvac system would provide an Application Programming Interface (API) for portions of data exchange, as well as the ability to deploy web servers to popular web services (Apache, IIS, etc.).

LITERATURE REVIEW

In this section, comprehensive literature reviews of existing evacuation modeling, vehicle routing optimization models, and solution algorithms are conducted. In addition, state-of-the-art transit systems in intermodal transportation and their implementation in emergency evacuation are presented.

Evacuation Modeling

Transit evacuation is playing an increasingly important role following the strikes of severe hurricanes such as Hurricanes Katrina and Rita in 2005, and more recently, Hurricane Sandy in 2012. To protect the general public from disaster, it is necessary to develop more advanced evacuation models for evaluating or optimizing transit evacuation operations. Most studies on transit evacuation operations focus on two types of off-line models: simulation models and optimization models. Simulation models are categorized into three groups: microscopic, macroscopic, and mesoscopic, depending on the level of detail at which the traffic information is described. Simulation models allow evacuation managers to develop and compare different evacuation plans for different hypothetical emergency scenarios (Yuan et al., 2006).

Macroscopic Simulation Model

Macroscopic simulation models consider traffic flow as composed of platoons of vehicles—i.e. vehicles with common characteristics are treated as a homogeneous group. They are mainly developed for evacuation planning purposes. Most macroscopic simulation models are based on a dynamic network flow approach (Sheffi et al., 1982; KLD, 1984; Hobeika and Jamei, 1985; Hobeika and Kim, 1998). In the context of emergency evacuation, macroscopic simulation models have uses such as analyzing traffic conditions, estimating evacuation times, and generating optimal evacuation routes.

NETVAC (Network Emergency Evacuation), developed by Sheffi et al. (1982), is considered to be the first evacuation planning simulation model. NETVAC is used for simulating traffic flow patterns and estimating clearance times during emergency evacuations. NETVAC allows the analyst to customize the degree of driver compliance on an intersection specific basis under evacuation conditions. NETVAC also supports dynamic route selection by dynamically adjusting the turning movements at each simulation interval according to the traffic conditions. However, this model was specifically designed for nuclear plant accident evacuation, which means the evacuation starts from a single point, and thus all the movements are directed radially outward from the single point, rather than in a more general direction as with hurricane evacuations.

HURREVAC (Hurricane Evacuation) (FEMA, 2013) is a storm tracking and decision support tool developed specifically for hurricane evacuation. HURREVAC combines the National Hurricane Center's Forecast Advisories with data from various state HES's (Hurricane Evacuation Studies) to estimate the time required to evacuate an area, which assists the local emergency management agency in determining the most appropriate evacuation decision time.

VISUM is a macroscopic simulation software system for traffic analyses. It is used to simulate evacuation plans (Schomborg et al., 2011; ARCADIS, 2011; ARCADIS, 2012), especially when the maximum evacuation time is required. ARCADIS Inc. (2011, 2012) performed VISUM simulations to forecast evacuation times in different scenarios. The VISUM network includes designated evacuation routes, as well as backup routes, to accurately reflect the traffic conditions during an evacuation. The potential impacts of the population growth on evacuation time were also analyzed.

Perkins et al. (2001) developed several models for hurricane evacuation planning to mitigate traffic congestion in North Carolina. Those models were used to determine total evacuation time, identify traffic bottlenecks, and assess traffic operation strategies. They also worked on the application of public transit in short-notice evacuation. An evacuation methodology which was specific to hurricane evacuation was developed to determine the scheduling of buses to evacuate the elderly and disabled citizens in North Carolina's small urban and rural areas. They assumed that buses were located at a single depot prior to evacuation. Each bus was assigned a location where it loaded the evacuees and proceeded to follow a predetermined route to reach the safe locations.

Microscopic Simulation Model

Microscopic simulation models focus on the modeling of individual vehicle behaviors and interactions among vehicles. Microscopic simulation models are generally based on car-following models. They are often used for modeling traffic with complex behavior in an emergency evacuation, such as contra-flow (Lim, 2003), traffic signal preemption (Zhang, 2009), and transit operations (Wen, 2012). Microscopic models are usually resource intensive, and thus, are only implemented in small networks.

CORSIM (Corridor Simulation) (McTrans, 2014) is a microscopic traffic simulation software package for simulating urban street and freeway traffic systems. It is an integration of two separate microscopic simulation models, NETSIM (Network Simulation) for modeling surface streets, and FRESIM (Freeway Simulation) for modeling freeways. NETSIM, which is the successor of UTCS-I (Urban Traffic Control System) in the 1970s, keeps track of each individual vehicle, including detailed characteristics relating to the vehicle within more complex urban networks. NETSIM provides simulation results in a more aggregated level. Lim (2003) and Theodoulou et al. (2004) utilized CORSIM to simulate hurricane evacuation with contra-flow strategy. Zou et al. (2005) applied CORSIM simulation technology to evaluate six plans for hurricane evacuations in Ocean City. Tagliaferri (2005) performed both CORSIM and VISSIM simulations to investigate the effects of the lane reversal plan on hurricane emergency evacuations. ORNL (Oak Ridge National Laboratory) (Bhaduri et al., 2006) developed OREMS (Oak Ridge Evacuation Modeling System), which is an integration of a CORSIM simulation model and a GIS model, to analyze and evaluate large-scale emergency evacuations, conduct evacuation time estimation, and develop evacuation plans. Zhang et al. (2009) proposed a CORSIM model for simulating emergency vehicle operations—including traffic signal preemption and movement on shoulder and red lights—in a hurricane evacuation. NETSIM is also capable of modeling transit operations. The impacts of transit signal priority and connected vehicles on transit emergency evacuations were also investigated. Wen et al. (2012) used CORSIM simulation with RTE to simulate transit signal priority and connected vehicles within a large network with over 150 signalized intersections.

Mesoscopic Simulation Model

Mesoscopic simulation models compromise between microscopic and macroscopic simulation models. They simulate individual vehicles with a high level of detail, but describe their activities and interactions based on aggregate relationships. The aggregation mitigates calculative burden and lessens computation time. Typical applications of mesoscopic simulation models in the context of emergency evacuation are reviewed as follows.

Dynasmart-P (Dynamic Network Assignment-Simulation Model for Advanced Roadway Telematics - Planning version), which is the planning version of Dynasmart (Mahmassani et al., 1994), utilizes mesoscopic models to represent traffic interactions. Dynasmart-P supports transportation network planning and operation analyses with a simulation-based dynamic traffic assignment. It is capable of handling a large-scale urban traffic network with up to 89,999 nodes (Mahmassani et al., 2004). In recent years, it has been enhanced to evaluate incident management strategies (Kwon, 2004; Yuan et al., 2006; Naser and Shawn, 2010). Kwon (2004) used Dynasmart-P simulations to evaluate emergency evacuation strategies on a large-scale network. Dynasmart-P simulations were developed for a hypothetical emergency evacuation in downtown Minneapolis, Minnesota. The model was calibrated using loop detection data. Alternative emergency evacuation strategies in terms of different network configurations were proposed and evaluated; however, the assumption that all drivers are aware of the network configuration changes and can adjust their routes accordingly is not realistic under actual emergencies. Naser and Shawn (2010) developed a Dynasmart-P application integrated with Cube-Voyager software (Citilabs, 2013), which provided an OD matrix for Dynasmart-P, to model flood evacuation at regional level. Different hypothetical emergency scenarios with varying flood locations, levels, and warning times were modeled using the Fargo-Moorhead metropolitan area data. Traffic controls were modified to facilitate the evacuation operations. The outputs of the Dynasmart-P simulation were used to estimate the evacuation time, measure the effectiveness of the modified traffic control, and evaluate the system parameters such as driver compliance and trip loading rate.

DynusT (Chiu et al., 2010) is another version of Dynasmart developed for real-time analysis that has been implemented in various evacuation studies (Chiu et al., 2008; Zhang et al., 2009; Zheng et al., 2010; Songchitruksa et al., 2012). Chiu et al. (2008) deployed and assessed the contra-flow operation in the Central Texas Evacuation network (CTE) in DynusT. The simulation results indicated that the contra-flow operation led to about a 14% travel time savings for all evacuees. Songchitruksa et al. (2012) created DynusT simulations for assessing the performance of alternative evacuation strategies, including partial contra-flow, as well as an "evaculane", in which evacuation traffic could use the outside paved shoulder as an additional traveling lane during an emergency evacuation, in the context of a hurricane evacuation in Houston, TX. Results indicated that the "evaculanes" on I-10 and US-290 could provide sufficient capacity to handle high evacuation demand on both routes without the contra-flow operation. In addition, the contra-flow plan for I-45 was proved adequate in handling high evacuation demand in lieu of fully implemented contra-flow operation. Wang et al. (2014) incorporated contra-flow with VMS (Variable Message Signs) in a hypothetical emergency evacuation. DynasT simulations were developed to evaluate the performance of the strategies. The simulation results demonstrated that the combination of contra-flow and VMS improved the evacuation performance more effectively than using only one or none of the two strategies.

Vehicle Routing Problem

The vehicle routing problem (VRP) was first introduced by Dantzig and Ramser (1959) as a generalization of the well-known "traveling salesman problem." The VRP involves finding a set of optimal routes for a fleet of vehicles to service a set of customers subjected to certain constraints. The classical VRP and its variants—such as the Capacitated Vehicle Routing Problem (CVRP), the VRP with time windows (VRPTW), and the VRP with Pickup and Delivery (VRPPD)—have been extensively studied for over 50 years. Current exact algorithms are able to solve the CVRP with a size limit of 50–100 customers depending on the customers' distribution and the response time requirement. However, in terms of the dynamic vehicle routing problem, most studies focus on heuristic algorithms, and no existing exact algorithms have been successfully applied to the vehicle routing problem in a transit emergency evacuation.

Exact Algorithms

Exact algorithms to solve the VRP include the branch-and-bound, the cutting plane, column generation, and the branch-and-price algorithms. A brief review of each of the exact algorithms is provided in this section.

The column generation algorithm is an efficient algorithm for solving large-scale linear programs. It has been widely applied to the VRP and its variants by many researchers. Agarwal et al. (1989), Hadjiconstantinou et al. (1995), and Bixby (1998) developed column generation algorithms for general VRP. Desrochers et al. (1992) applied a column generation algorithm on the VRPTW. Jin et al. (2008) proposed a column generation approach to solve the VRP with split delivery (VRPSD). The basic idea of column generation is to iteratively generate a subset of columns and push them into the basis such that the inclusion potentially improves the objective function. The column generation algorithm can be combined with the branch-and-bound algorithm, also called the branch-and-price algorithm. The branching occurs when no columns can enter the basis, and the solution to the linear program is not an integer.

Desrochers et al. (1992) presented a dynamic programming-based optimization algorithm for the VRPTW. The VRPTW is formulated by a Set Covering (SC) form in which the path does not have to be elementary. The LP relaxation of the SC model is solved by column generation. The pricing sub-problem, which is the Shortest Path Problem with Resource Constraints (SPPRC), is solved through a label correcting algorithm in which labels are created through a "pulling" process. Two sets of labels were generated for the states at each node. The first set of labels provides an upper bound, while the second set of labels relates to a lower bound on the cost of a path associated with a state at each node. The algorithm computes the cost associated with a state at a node by progressive refinement of lower and upper bounds on its value. In addition, a 2-cycle elimination procedure was accomplished by a duplication of the labels. This procedure could tighten the relaxed state space by eliminating all cycles of length two. The LP solution is then used in a branch-and-bound algorithm to solve the integer SC model. The algorithm has a pseudo polynomial complexity.

Feillet et al. (2004) proposed an exact algorithm for the Elementary Shortest Path Problem with Resource Constraint (ESPPRC). The algorithm is adapted from Desrochers' (1988) label correcting algorithm. A new resource, which indicates if a label of a node is extendable to another node, is created to enforce the elementary path constraint, as proposed by Beasley and Christofides (1989). The label correcting method is improved by introducing the new resource in the dominance rule. This method could decrease the number of states to be explored, therefore reducing the computational complexity. The drawback of the method is that the complexity is strongly related to the graph structure, the number of the nodes, and the tightness of resource constraints.

Righini and Salani (2008) developed a label setting algorithm for the ESPPRC. The traditional label setting algorithm is improved by two new methods. The first method is a bi-directional search with resources bounding in which states are extended both forward from a start node to its successors and backward from a destination node to its predecessors. All the forward states and backward states at a node are then joined, subject to resource constraints, to make feasible routes. Therefore, states are not extended if, at most, half of the available amount of resources has been used. This method could effectively reduce the number of states in the solution space. The second method is a combination of bi-directional search with state space relaxation. In this algorithm, the state space is relaxed to allow cycles with length more than two. The path found from the relaxed state space is guaranteed to be feasible with regards to the resource constraints, but it is not guaranteed to be elementary. Righini and Salani also provided branch-and-bound strategies to eliminate cycles in order to solve the ESPPRC to optimality.

The pricing sub-problem in the column generation scheme was also called the Traveling Salesman Problem with Profits (TSPP) by Feillet et al. (2005) in a comprehensive survey. TSPP is considered as a bi-criteria TSP with two opposite objectives: to maximize the benefits collection at each vertex, which pushes the salesman to travel; and to minimize the travel cost, which prevents the salesman from traveling. The two objects constitute the price of visiting a vertex. Generally, TSPP is divided into three categories based on the way the two objectives are presented: (1) Profitable Tour Problem (PTP) by (Dell'Amico et al., 1995), in which both objectives are combined in the objective function; (2) Orienteering Problem (OP) by (Golden et al., 1987), in which the travel cost is formulated as a constraint; and (3) Prize-Collecting TSP (PCTSP) by (Balas, 1989), in which the profit is stated as a constraint. Solution approaches for TSPP were summarized into three groups: (1) exact algorithms; (2) classical heuristics; and (3) meta-heuristic procedures. The performance and applicability of the approaches were identified for different TSPP applications.

Liu and Lee (2003) considered stochastic customer demand and included inventory costs in the vehicle routing problem. An initial solution was found by clustering the customers based on an increasing order of their marginal inventory costs. For each cluster, the depot was located nearest to the center, and a travelling salesman problem was solved. Then, a hierarchical improvement method was used based on the moves, drops, and shifts of the depot's location. Both routing and inventory costs were fully evaluated for possible moves; however, the procedure was much slower than nested methods used in route length estimation.

Heuristics

Goel and Gruhn (2005) worked on a real-life vehicle routing problem with randomly generated demands after the start of planning. They considered a diversity of practical constraints, such as time window restrictions, a heterogeneous vehicle fleet, vehicle compatibility constraints, etc. To cope with the complexities of the problem, they improved the Large Neighborhood Search method by using fast insertion methods as the search algorithm. Two insertion methods were developed. The first is a sequential insertion method in which unscheduled transportation requests were randomly chosen, and all feasible insertion possibilities were considered. The second is an auction method in which the vehicles only considered were unscheduled transportation requests with low incremental costs. The second method was used for the vehicle routing with time windows.

Schwardt and Dethloff (2005) developed a variant of Kohonen's algorithm to solve a deterministic, single-depot, capacitated multi-vehicle routing problem. Kohonen's algorithm was based on a neural network including two layers. Weights, which were Euclidean distances between nodes and customer demands, were assigned to the links between the layers. The neural network used self-organization approaches to construct the vehicle mappings and simultaneously generate feasible solutions to the location-routing problem.

Dynamic Vehicle Routing

Secomandi (2000) worked on a single-vehicle routing problem in which customers' demands are uncertain and are assumed to follow certain distributions. The problem was formulated as a stochastic shortest path problem (SSPP) in a discrete-time dynamic system. He developed two sets of algorithms based on Neuro-Dynamic Programming (NDP). NDP is an emerging field of neural network application. Unlike traditional neural networks, where training data sets are a necessity before training starts, NDP creates the training data set while it is running. Moreover, NDP trains data repeatedly throughout the optimization process by using linear training architecture. The first set is called the optimistic approximate policy iteration (OAPI), in which the cost-to-go function is approximated by a linear equation with a vector of parameters. At each time step, the training data set—which contains a certain number of possible solutions under current vehicles routing states and its corresponding cost-to-go values—is collected by simulation. Then, a least-square fitting is performed on the training set in order to generate the parameters of the linear function for all possible solutions under current vehicles' routing states. Finally, the current vehicles' states can be optimized and go to the next state by a greedy algorithm with respect to the approximated cost-to-go function under the current state. The second set is called rollout policies (RP), which is a variant of OAPI. RP sequentially adjusts the decisions under current vehicles routing states by employing the cyclic heuristic. As opposed to the OAPI, RP approximates the cost-to-go function by a closed form function where the parameters of the form are trivially trained to be zero and one at each step.

Afshar and Haghani (2008) developed a simulation-optimization framework to optimize evacuation operations. The framework consists of two parts: a traffic simulator and an optimization module. The simple traffic simulation was programmed based on the Green shield's model. The traffic simulator provides time-varying link travel times for the optimization module and facilitates the performance measures calculation. The optimization module simultaneously optimizes the total evacuees' travel time and the network clearance time. Two classes of modules in the optimization framework were proposed. The first module, named "SPREAD", was used to distribute transit vehicles' departure over a long time to relieve congestion effects in the network. The second module, named "SQUEEZE", was used to consolidate the evacuees' distribution along time intervals to minimize the last departure time. Since these two objectives are contradictory, the system optimal solution seeks a trade-off between these two objectives.

Web-based Transit Service

This section contains several widely studied and utilized trip planning systems. The web-based transit service architecture will be discussed in the next part of this section. Several online trip planning systems were developed and published up to twenty years ago. In the research of Peng and Huang (2000), it was determined that an online trip planning system should include two different aspects: information content and system functionality. The information content contains basic information about service, static information for transit, trip planning information, and real-time information. The system functionality includes information dissemination, interactive communication, and online transactions. Many existing online trip planners focus on providing interactive static transit information. Google Transit Trip Planner, which is based on the Google Maps platform, provides comprehensive and powerful functionalities that contain directions for travel, travel time, and transfer information. Google also provides different mode options including bus, train, subway, and light rail. The function allows the user to choose their favorite or most convenient transport methods. Also, Google builds in three optimization models into their transit planning tool: best route, fewer transfers, and less walking. Cherry et al. (2006) provided another transit online trip planner. In their research and development, online trip planning could support three origin and destination selection methods: typing in the text address, selecting landmarks, and clicking on locations on the map that corresponded with Google Maps functions. Furthermore, a forward-searching algorithm was implemented and executed from the origin to destination, or alternately from destination back to the origin. The animation of this online trip planner was integrated with commercial GIS software—ArcGIS and ArcIMS—which provided professional and accurate map information to users. Some other online trip planning systems were discussed by Sun et al. (2011). In their research, Sun et al. listed map-based interactive online trip planning systems, which included South-East Wisconsin Transit Trip Planner, ATIS, Transport Direct in England, and ENOSIS. All of the online trip planning systems are integrations of map information and transit optimization algorithms. Also, Sun et al. discussed the architecture in the online trip planning system and suggested the use of Service-Oriented Architecture (SOA) for development web-based transit service. Figure 1 displays the typical structure of online trip planning service.

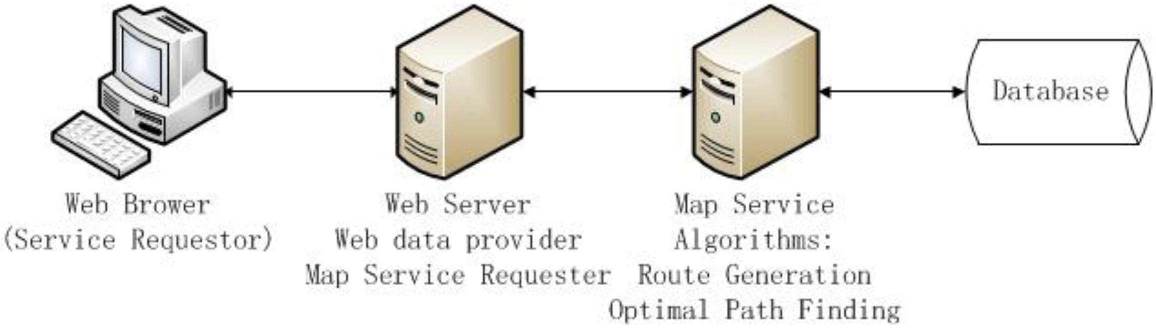


Figure 1 Online Transit Planning System Architecture

Hoar (2008) and Brian et al. (2009) stated that their visualizing transit system used the same architecture for development. SOA, which is a standard framework for integrating existing distributed resources, was to reuse the components (services) within the existing systems and provide interoperability among modules that were built with different nine programming tools or even across platforms.

Transit in Intermodal Transportation

In this section, the transit in intermodal transportation is discussed. Several studies suggested that effective schedule coordination may significantly increase the attractiveness and productivity of linear intermodal transit systems (Chowdhury and Chien, 2000). In their subsequent research, Chowdhury and Chien (2001) provide a model for coordinating a multimodal transit system. They found that coordination is preferable if train and bus headways are large. The assumptions of their optimization model considered locations of transit facilities, supplier side, demand side administration costs, vehicle maintenance costs, insurance costs, labor costs, energy consumption costs, bus arrivals, and the probability of a late vehicle arrival time. The formulated total cost function was used when optimizing the transfer coordination. The cost parameters contained supplier costs, user costs, wait costs, transfer costs, and in-vehicle costs. The minimized two-stage procedure included Stage One, in which the optimal headways of all routes were determined without coordination; and Stage Two, in which train-bus coordination at all transfer stations were considered. In Stage Two specifically, the procedure for optimizing coordinated operations in intermodal systems is an integration of train-to-bus and bus-to-train demand, bus routes without coordination, and train transfer stations. Furthermore, research by Chowdhury and Chien (2011) showed that decision variables in bus-train coordination could include bus size and slack time considering the synchronization of bus and train arrivals. Overall, this direction of transit in intermodal transportation systems was minimizing total cost with/without coordination in two stages. Other researchers (Graham et al, 2000 and Cassady et al 2004) focused on different directions of an intermodal transportation system. In their intermodal coordination system, the intermodal facility model terminal performance had more proportion. The coordination model of their studies paid more attention to cargo transportation than transit passenger transportation. Another study by Chan (2010) proposed a two-stage model for carless evacuation. This proposal included a location problem that aimed at congregating the carless at specific locations and a routing problem with the objective of picking up the carless from evacuation sites and delivering them to safe areas. They explicitly considered the dynamic demand pattern of evacuees to pickup points, as well as multiple trips of buses from pickup points to shelters. Yue Liu and Jie Yu (2011) reported in the transit-based evacuation; most of the studies have not integrated the dynamic processes of evacuee guidance (from buildings or parking lots to pickup points) and bus routing (from pickup points to shelters). Their work was an integrated optimization model that is capable of coordinating the evacuee guidance and transit routing process seamlessly and simultaneously. Their model was a two-stage model that was formulated as a combined vehicle routing and assignment problem, and it was solved by a two-stage Tabu-based heuristic to yield meta-optimal solutions. The feasibility and applicability of the proposed model were illustrated with a numerical example solved to optimality. Results showed that the proposed model can yield valid and detailed evacuee guiding and transit routing plans during the evacuation within a reasonable time window.

METHODOLOGY

In this chapter, development of a Capacitated Dynamic Vehicle Routing Problem with Pickup and Delivery model in the context of an emergence evacuation is discussed. Solution approaches to the CDVRPPD model are presented. The computational results on benchmark problems are compared to the results from other studies in the literature.

Problem Statement

Given a transportation network in which emergency evacuation is carried out, the dispatching of transit services resembles the Capacitated Dynamic Vehicle Routing Problem with Pickup and Delivery. The transit emergency evacuation process includes sending transit vehicles from the Coast Transit Authority (CTA) to the hurricane prone area to pick up evacuees, updating transit vehicle routes based on real-time evacuee and traffic information, and delivering evacuees to the designated shelters. The evacuee and traffic information are dynamic in nature. The CTA provides a dial-a-ride service that allows evacuees to call in requesting on-site pickup during the evacuation process. The dispatcher has no knowledge of future pickup requests. The information of a real-time pickup request, including its location and demand, become known from the moment it comes into the system. From here, the problem is assigning the most appropriate vehicle to the new request. Routes are formed before evacuation but updated dynamically in response to real-time information updates, including new pickup requests and travel time changes. Comparing to the static vehicle routing problem in which demands are known before the evacuation, the dynamic feature gives more freedom to the evacuees while, at the same time, bringing more challenge to transit agencies.

Model Development

The problem described above can be formulated as a special case of Capacitated Dynamic Vehicle Routing Problem with Pickup and Delivery. It consists of two types of problems: a) A static CVRPPD (Capacitated Vehicle Routing Problem with Pickup and

Delivery) in the planning stage of an emergency evacuation; and b) A CDVRPPD after the emergency evacuation starts.

Base Model

In the planning stage of an emergency evacuation, pre-registered evacuees' information, including their demands and locations, are known in advance. Despite the dynamic factors, such as travel time fluctuations, the CVRPPD is assumed static in this stage. The classical CVRPPD generalizes the traveling salesman problem; thus it is NP-hard. The CVRPPD is defined on a directed graph $G = (V, A)$, where V is the set of vertices, and A is the set of arcs. S denotes the set of depots. N denotes the set of pickup points and M denotes the set of shelters, both of which are considered as customers with pickup and delivery demands. Therefore, the graph consists of $|N|+|M|+|S|$ vertices such that $V = N \bigcup M \bigcup S$.

The set of arcs, A, represents direct connections among the vertices. A non-negative cost c_{ij} is assigned to each arc $(i, j) \in A$. Arc cost c_{ij} generally represents the travel time going from vertex i to vertex j, which corresponds to the shortest path from vertex i to vertex j, and consequently the cost matrix satisfies the triangle inequality. Satisfying the triangle inequality, $c_{iz} + c_{zj} \geq c_{ij}$ for all $i, j, z \in V$, implies that any removal of pickup requests from a feasible route will reduce the route cost, and any insertion of pickup requests to a feasible route will increase the routes cost. Using self-loop is not allowed by imposing $c_{ii} = +\infty$, for all $i \in V$. The graph is directed with an asymmetric cost matrix. This is realistic, especially in the case of an emergency evacuation where outbound traffic is usually much heavier than inbound traffic.

There are certain restrictions imposed on the graph as shown in Figure 2. The restrictions are written in the form of $i \not\rightarrow j$, where i and j denote the nodes which constitute a restricted link. For example, an $S \not\rightarrow M$ restriction indicates that a vehicle cannot travel from a depot to a shelter, which means that a route has to pass at least one pickup point before ending at a shelter. Other restrictions—including $M \not\to M$, which denotes a vehicle cannot travel among shelters, $M \not\to N$, which denotes a vehicle cannot travel from a shelter to a pickup point, and $N \not\to S$, which denotes a vehicle cannot travel from a pickup point to a depot—are added according to real emergency evacuation situations. These restrictions theoretically turn the network into an incomplete graph where some of the arcs are restricted. These restrictions can be symbolized by assigning 0 to the decision variables corresponding to the usage of the restricted arcs in the model. these restrictions are represented alternatively to impose a very large positive value to the travel cost on restricted arcs. In this report, the latter is used because it is easier to implement.

Each pickup point i, $i \in N$, has a deterministic non-negative demand d_i. It is assumed that d_i is less than or equal to the vehicle capacity. If d_i is larger than the vehicle capacity, the pickup point i will be divided into multiple pickup points, which coincide with and have less demand than the vehicle capacity. For depots and shelters, their demands are fictitiously set to 0. There is also a fixed service time c_s associated with each pickup point i. The service time represents the time needed for loading and unloading and is included in the travel cost c_{ij} associated with each arc. Based on CTA's experience, the service time at a pickup point is normally 2.5 minutes on average. The capacity of each shelter is assumed to be unlimited, which conforms to the actual situation in the Mississippi Gulf Coast. Therefore, any one of the shelters can accommodate all the evacuees in the network.

A homogenous fleet of transit vehicles K with identical capacity Q services the pickup points and shelters. The fleet size is infinite. Q must be larger than or equal to the sum of all the demands on the route assigned to vehicle k. Overload is not permitted, and each pickup point is serviced exactly once. The service includes scheduled pickup for registered evacuees, dial-a-ride to unregistered evacuees, and delivery to a designated shelter. A precedence constraint which regulates that all the pickup points must be served before any shelter is imposed on the route. The CVRPPD involves the design of a set of minimum cost routes that originate at a depot in *S* and terminate at a shelter in *M* after picking up all the evacuees. Practically, a vehicle does not have to be back to the depot immediately after its arrival at a shelter. However, in order to form a complete route, a set of dummy arcs linking from the shelters to the depots with zero travel cost are introduced to replace the original arcs. Then, each vehicle can go back to the depot after delivery at a shelter via the dummy link. In this case, a directed cycle is associated with a vehicle route.

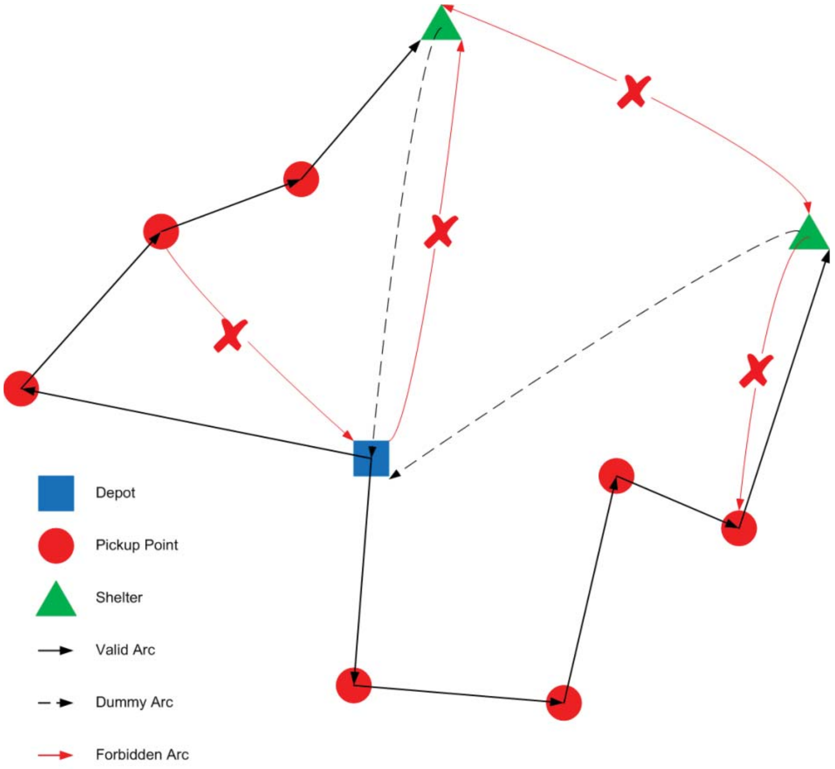


Figure 2 A Simple Representation of Vehicle Routes

Figure 2 shows a sample of vehicle routes in the network. The blue square denotes a depot. A vehicle starts from the depot and then picks up evacuees at the pickup points, which are represented by the red dots. After pickup, the vehicle will deliver the evacuees to a shelter, which is represented by the green triangle. The black solid line with an arrow denotes the arc that forms a vehicle route. The black dotted line with an arrow denotes the dummy link that connects a shelter to a depot. There is no cost associated with a dummy link. The red solid line with an arrow denotes the arc that is restricted in the model.

For ease of reference, notations are summarized as follows. Particularly, the set of depots, S, contains only one element since the scope of this study is to solve the CVRPPD with single depot.

G = A graph represents the transportation network.

V = Set of vertices in G.

A = Set of arcs in G.

M = Set of shelters, $M = \{1, 2, \ldots, m\}$.

N = Set of pickup points at the beginning of evacuation, $N = \{m+1, m+2, \ldots, m+n\}$.

S = Set of depots of all vehicles, $S = \{0\}$.

K = Set of a fleet of vehicles, $K = \{0, 1, 2, \ldots, k, \ldots\}$.

c_{ij} = Travel cost, $\forall (i, j) \in A$.

d_i = Demand at pickup point i, $\forall i \in N$.

u_i^k = Vehicle k's load after visiting pickup point i, $\forall i \in N$, $\forall k \in K$.

Q = Vehicle capacity.

The CVRPPD is mathematically formulated as an integer linear programming model by (3.1) – (3.12). The set of decision variables is defined as x_{ij}^k. For each arc $(i, j) \in A$, the integer variable x_{ij}^k indicates whether (i, j) is traversed by vehicle k in the solution.

$$x_{ij}^k = \begin{cases} 1, \text{ if vehicle } k \text{ travels directly from vertex } i \text{ to vertex } j, \\ \quad \forall i \in V,\ \forall j \in V,\ i \neq j,\ \forall k \in K \\ 0, \text{ otherwise} \end{cases}$$

$$\min z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V \setminus i} c_{ij} x_{ij}^k \tag{3.1}$$

$$\sum_{k\in K}\sum_{j\in V\setminus S} x_{ij}^{k} = 1,\ \forall i \in N \tag{3.2}$$

$$\sum_{i\in V} x_{iz}^{k} - \sum_{j\in V} x_{zj}^{k} = 0,\ \forall z \in V,\ z \neq i,\ z \neq j,\ \forall k \in K \tag{3.3}$$

$$\sum_{j\in N} x_{ij}^{k} = 1,\ i \in S,\ \forall k \in K \tag{3.4}$$

$$\sum_{i\in N}\sum_{j\in M} x_{ij}^{k} = 1,\ \forall k \in K \tag{3.5}$$

$$\sum_{i\in M} x_{ij}^{k} = 1,\ j \in S, \forall k \in K \tag{3.6}$$

$$x_{ij}^{k} = 0,\ i \in S, \forall j \in M, \forall k \in K \tag{3.7}$$

$$x_{ij}^{k} = 0,\ j \in S, \forall i \in N, \forall k \in K \tag{3.8}$$

$$x_{ij}^{k} = 0,\ \forall i \in M, \forall j \in N \bigcup M, \forall k \in K \tag{3.9}$$

$$u_{i}^{k} - u_{j}^{k} + Qx_{ij}^{k} \leq Q - d_{j},\ \forall i, j \in N,\ i \neq j,\ \forall k \in K,\ such\ that\ d_{i} + d_{j} \leq Q \tag{3.10}$$

$$d_{i} \leq u_{i}^{k} \leq Q,\ \forall i \in N,\ \forall k \in K \tag{3.11}$$

$$x_{ij}^{k} \in \{0,1\},\ \forall i, j \in V, \forall k \in K \tag{3.12}$$

The objective function (3.1) is to minimize the total travel cost. The in-degree constraints (3.2) ensure that each pickup point is visited once and only once. Route continuity is enforced by the constraints (3.3), as once a vehicle arrives at a pickup point, it has to leave the pickup point. The constraints (3.4), (3.5), and (3.6) indicate that each vehicle leaves an depot exactly once; after picking up all the evacuees on its route, it has to visit a shelter once and only once; and finally travels back to the depot, respectively. The constraints (3.7), (3.8), and (3.9) are connectivity constraints indicating that the arcs from the a depot to a shelter, arcs from a shelter to a pickup point, arcs among shelters, and arcs from a pickup point to a depot are restricted, respectively. The constraints (3.10) and (3.11) are called polynomial cardinality constraints (Christofides et al., 1979) that impose both sub-tour elimination and the vehicle capacity requirements. Constraints (3.12) are the integrality constraints.

Column Generation Model

In the base model, only the in-degree constraints (3.2) are correlated with the vehicles while the remaining constraints are associated with individual vehicles separately. Since the number of constraints (3.3) – (3.12) is exponential, it is desirable to reformulate the integer program into another equivalent problem which is more manageable for the simplex method. One of the most successful methods is Dantzig-Wolfe decomposition (1960) which breaks up the base model into a master problem and sub-problems. The decomposition actually decreases the number of constraints, but increases the number of variables exponentially. For large scale IP, the decomposed model is too large to consider all the variables explicitly. Since for the master program solved by the simplex algorithm, most columns are inactive at each step. In such a scheme, a column generation approach, which represents a generalized application of Dantzig-Wolfe decomposition, is proposed to solve large integer problems by working with only a subset of variables.

Master Problem Model

The column generation is based on a master problem and sub-problem structure. The master problem is an integer problem which is usually relaxed to a linear problem that is easier to solve.

For the CVRPPD problem defined through (3.1) – (3.12), the constraints (3.2) are considered to be the linking constraints in a Dantzig-Wolfe decomposition scheme, which connects the vehicle routes, while the remaining constraints (3.3) – (3.12) are associated with individual vehicle. The constraints (3.3) – (3.12) define the domain of individual vehicle route generation, which is the sub-problem. Let R^k be the set of feasible routes traveled by vehicle k, while r represents an elementary route in R^k. Let x_{ijr}^{k} be a binary variable defined as follows.

$$x_{ijr}^{k} = \begin{cases} 1, \text{ if vehicle } k \text{ travels directly from } i \text{ to } j \text{ on path } r, \\ \quad \forall i \in V,\ \forall j \in V,\ i \neq j,\ \forall r \in R^{k},\ \forall k \in K \\ 0, \text{ otherwise} \end{cases}$$

Each variable x_{ij}^{k} in the base model can be represented by a combination of x_{ijr}^{k}. The decision variable x_{ij}^{k} is rewritten by (3.13) – (3.15).

$$x_{ij}^{k} = \sum_{r \in R^{k}} x_{ijr}^{k} y_{r}^{k},\ \forall k \in K,\ \forall i \in V,\ \forall j \in V \tag{3.13}$$

$$\sum_{r \in R^{k}} y_{r}^{k} = 1,\ \forall k \in K \tag{3.14}$$

$$y_{r}^{k} \in \{0,1\},\ \forall r \in R^{k},\ \forall k \in K \tag{3.15}$$

Where, y_{r}^{k} is binary variable that represents whether vehicle *k* travels on path *r*. The cost of route *r*, c_{r}^{k}, and the number of times a pickup point *i* is visited by vehicle *k* on route *r*, a_{ir}^{k} are defined as,

$$c_{r}^{k} = \sum_{i,j \in V} c_{ij} x_{ijr}^{k},\ \forall r \in R^{k},\ \forall k \in K \tag{3.16}$$

$$a_{ik}^{r} = \sum_{j \in V \setminus i} x_{ijk}^{r},\ \forall r \in R^{k},\ \forall k \in K,\ \forall i \in V \tag{3.17}$$

Substitute x_{ij}^{k} and c_{ij} in (3.1) and (3.2) using (3.16) – (3.17). The reformulated model is shown by (3.18) – (3.21).

$$\min \sum_{k\in K}\sum_{r\in R^k} c_r^k y_r^k \tag{3.18}$$

$$\sum_{k\in K}\sum_{r\in R^k} a_{ir}^k y_r^k = 1,\ \forall i \in N \tag{3.19}$$

$$\sum_{r\in R^k} y_r^k = 1,\ \forall k \in K \tag{3.20}$$

$$y_r^k \in \{0,1\},\ \forall r \in R^k,\ \forall k \in K \tag{3.21}$$

Since the fleet of vehicles is homogenous, the travel cost is only associated with the arc, such that $c_r = c_r^k$ for all vehicle *k*. The route sets $R^k = R$ for all vehicles *k*. Therefore, it is possible to eliminate the index *k* by aggregating vehicle *k*'s parameters on route *r*. The revised model is presented as follows.

$$MP : \min \sum_{r\in R} c_r y_r \tag{3.22}$$

$$\sum_{r\in R} a_{ir} y_r = 1,\ \forall i \in N \tag{3.23}$$

$$y_r \in \{0,1\},\ \forall r \in R \tag{3.24}$$

Now, equations (3.22) – (3.24) constitute the master problem of the SP model. Notation a_{ir} is binary variable that equals to 1 if vertex *i* is visited by route *r* and equals to 0 otherwise. Decision variable y_r is a binary variable that equals to 1 if route *r* is used in the optimum solution and equals to 0 otherwise. Constraint (3.35) states that each pickup point *i* is covered by one and only one route *r* in the routes set *R*. The master problem is usually relaxed to a Linear Master Problem (LMP) by replacing the integrality constraint (3.36) with $y_r \in [0,1],\ \forall r \in R$. The columns represented by the decision variables correspond to the feasible routes. Since the number of columns, |*R*|, exponentially increases with the problem size, it is not practical to explicitly enumerate all feasible routes and solve the master problem as an integer programming problem for all but very small sized problems. For example, a network with *n* customers has theoretically *e*(*n*!) elementary routes when *n* is sufficiently large. The appealing idea to overcome this difficulty is to work with only a small subset of variables first and then generate new variables as needed. The master problem that considers only a subset of variables is so called Restricted Master Problem (RMP). The linear relaxation of the RMP (LRMP) is represented by (3.25) – (3.28). The special structure of the SP model results in a tighter linear programming relaxation than that of the arc-based CVRPPD model.

$$LRMP: \min \sum_{r \in R'} c_r y_r \tag{3.25}$$

$$\sum_{r \in R'} a_{ir} y_r = 1, \ \forall i \in N \tag{3.26}$$

$$y_r \in [0,1], \ \forall r \in R' \tag{3.27}$$

$$R' \subseteq R \tag{3.28}$$

Where, R' is a subset of *R*. The objective of the RMP is to find a set of optimum cost routes within R' to service the pickup points. In the form of a linear relaxation of the RMP, each decision variable $y_r, \ \forall r \in R'$ represents the number of times the path *r* is used in the optimum solution. The decision variable y_r is not necessarily an integer; it is actually possible for the decision variable to be any real number in the interval [0, 1].

Instead of the SP model, in which each pickup point is visited exactly once, Desrochers et al. (1992) presented a Set Covering (SC) model which no longer requires the routes in *R* to be elementary. In the SC model, a_{ir} represents the number of times a pickup point *i* is visited by route *r*. It can take any positive integer values, not just a binary value. Therefore, a new constraint (3.29) is proposed to replace constraint (3.26).

$$\sum_{r \in R'} a_{ir} y_r \geq 1, \ \forall i \in N \tag{3.29}$$

Although the relaxation of (3.26) yields a weaker lower bound than that of the SP model because of the existence of non-elementary routes in *R*, the SC model is still more beneficial than the SP model. First, the SC model is numerically more stable than the SP model especially in the environments involving many customers on the same route (Desrochers et al., 1992). Second, the linear relaxation of the SC model is easier to solve than the SP model (Jin et al., 2008).

The SC model described above is very general but can be easily extended to other CVRPPDs with variants, such as time windows or priority queues. Constraints with regard to the special requirements are applied to the sub-problem to generate a feasible route.

Since the number of all feasible routes in a CVRPPD instance increases exponentially with the problem size, explicitly enumerating all the feasible routes is not an option for a large size CVRPPD. Therefore, the column generation based approach is applied to solve the problem. One of the key steps in column generation is to design a sub-problem model for generating columns into *R'* so that *R'* is expanded progressively towards the optimum solution.

Sub-Problem Model

Every linear programming problem has an associated dual linear programming problem. For the CVRPPD, the LRMP is referred to as a primal problem. Let $y_r = \left\{ y_1, y_2, \ldots, y_{|R'|} \right\}$ be the optimal solution to the LRMP. It is necessary to identify whether y_r is also an optimum solution to the LMP.

Let $\delta = \{\delta_1, \delta_2, \ldots, \delta_n\}$ be the set of dual variables associated with (3.29) and $\delta = \left\{\delta_1, \delta_2, \ldots, \delta_n\right\}$ be the dual optimal solution with respect to y_r. The dual of the linear relaxation of the master problem (LMPD) is represented as follows.

$$LMPD : \max \sum_{i \in N} \delta_i \tag{3.30}$$

$$\sum_{i \in N} a_{ir} \delta_i \leq c_r, \ \forall r \in R' \tag{3.31}$$

$$\delta_i \geq 0, \ \forall i \in N \tag{3.32}$$

Clearly δ satisfies constraints (3.31) for all $r \in R'$. Hence if we can prove that δ satisfies constraints (3.31) for all $r \in R$, δ is optimum for the LMPD and thus y_r is optimum for the LMP according to the duality theorem (Boyd et al., 2009). Instead, if there is a route r, $r \in R$ that violates the constraints (3.31), the current δ is not optimum for LMPD. The corresponding route r, which causes the violation, can be added into R' of the LRMP. The LRMP is then solved again. This process repeats until no route violating constraints (3.31) can be found (See Figure 3). At this point, the optimum solutions, y and δ, are found for the LMP and LMPD, respectively.

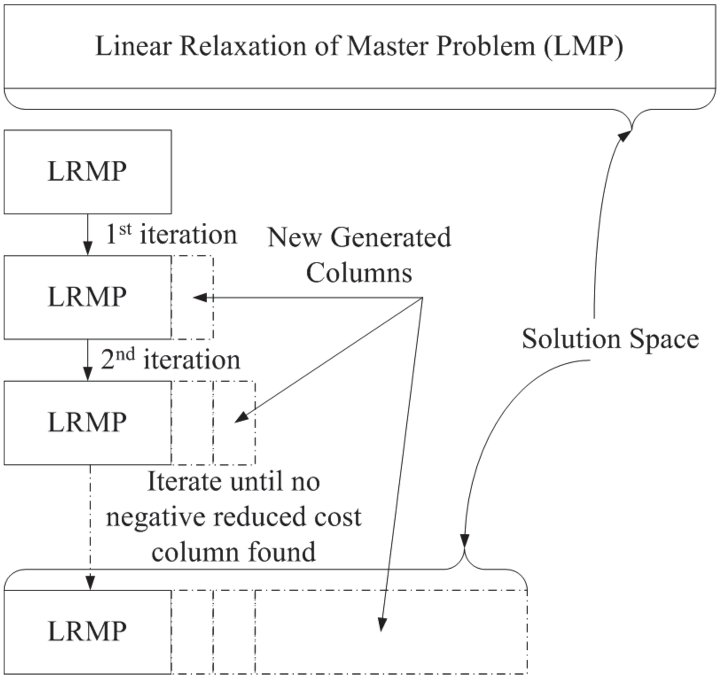

Linear Relaxation of Master Problem (LMP)
LRMP
1st iteration
New Generated Columns
LRMP
2nd iteration
Solution Space
LRMP
Iterate until no negative reduced cost column found
LRMP

Figure 3 Columns Set Augmentation

Figure 3 illustrates the relationship between the LRMP and the LMP in terms of the number of columns. The first row of Figure 3 shows the complete set of columns. The rest of rows demonstrates how the columns set is augmented towards the optimum solution for each iteration of the column generation process.

Let c_r be the reduced cost of a route *r*. c_r is formulated as follows.

$$c_r = c_r - \sum_{i \in N} a_{ir}\delta_i, \ \forall r \in R \qquad (3.33)$$

The sub-problem now is to find a feasible route *r* with negative c_r. The sub-problem must be able to efficiently price out all feasible routes, that is the reason it is usually called pricing problem. Then, the sub-problem decomposes into *n* identical problems, each of which is an Elementary Shortest Path Problem with Capacity Constraint (ESPPCC) defined on the same graph as the master problem. The ESPPCC model is formulated as follows.

$$\min \sum_{i \in V} \sum_{j \in V \setminus i} c_{ij} z_{ij} \qquad (3.34)$$

$$\sum_{i \in M} \sum_{j \in S} z_{ij} = 1 \tag{3.35}$$

$$\sum_{i \in N} \sum_{j \in M} z_{ij} = 1 \tag{3.36}$$

$$\sum_{i \in S} \sum_{j \in N} z_{ij} = 1 \tag{3.37}$$

$$\sum_{i \in V} z_{io} - \sum_{j \in V} z_{oj} = 0,\ \forall o \in V,\ o \neq i,\ o \neq j \tag{3.38}$$

$$z_{ij} = 0,\ \forall i \in S, \forall j \in M \tag{3.39}$$

$$z_{ij} = 0,\ \forall i \in N, \forall j \in S \tag{3.40}$$

$$z_{ij} = 0,\ \forall i \in M, \forall j \in N \bigcup M \tag{3.41}$$

$$u_i - u_j + Qz_{ij} \leq Q - d_j,\ \forall i, j \in N,\ i \neq j,\ such\ that\ d_i + d_j \leq Q \tag{3.42}$$

$$d_i \leq u_i \leq Q,\ \forall i \in N \tag{3.43}$$

$$z_{ij} \in \{0,1\},\ \forall i, j \in V \tag{3.44}$$

Where,

c_{ij} = Cost of using arc *(i, j)*, where $c_{ij} = c_{ij} - \frac{\delta_i}{2} - \frac{\delta_j}{2}$

z_{ij} is the decision variable that represents flow in the network.

$$z_{ij} = \begin{cases} 1, \text{ if arc } (i, j) \text{ is used in the shortest path}, \ \forall i \in V,\ \forall j \in V,\ i \neq j \\ 0, \text{ otherwise} \end{cases}$$

The objective is to find the shortest path with a negative reduced cost that covers a subset of pickup points $N', N' \subseteq N$. Constraints (3.35) – (3.38) are flow conservation constraints. Constraints (3.39) – (3.41) are connectivity constraints. Constraints (3.42) –

(3.43) are the sub-tour elimination constraints, where u_i is the vehicle load after visiting pickup point *i*. Constraint (3.44) ensures the integrality.

Since the ESPPCC is NP-hard (Dror, 1994), allowing cycles on the shortest path by relaxing some of the constraints—which changes the ESPPCC to the non-elementary Shortest Path Problem with Capacity Constraint (SPPCC) (Desrosiers et al., 1992; Irnich and Villeneuve, 2006)—becomes imperative regarding the computational burden. However, allowing cycles on the shortest path will expand the columns set *R* and thus provide a weaker lower bound to the master problem. Therefore, researchers focused on compromising between complexity and quality. Beasley and Christofides (1989) imposed a new resource on each node indicating the vertices that has been previously visited so as to prevent cycles. Desrochers et al. (1992) provided a 2-cycle elimination algorithm that eliminates the cycles with an *i*-*j*-*i* form. Irnich and Villeneuve (2006) extended the 2-cycle elimination to *k*-cycle elimination where cycles containing *k* (or less) nodes are removed.

CDVRPPD Model

In a real-time scheme, a planning horizon [0, *H*] is applied to the evacuation process as illustrated in Figure 4, where *H* is the maximum evacuation time. *H* is evenly divided into $\lfloor H/l \rfloor$ intervals with equal length *l*. The length *l* is determined based on the problem size.

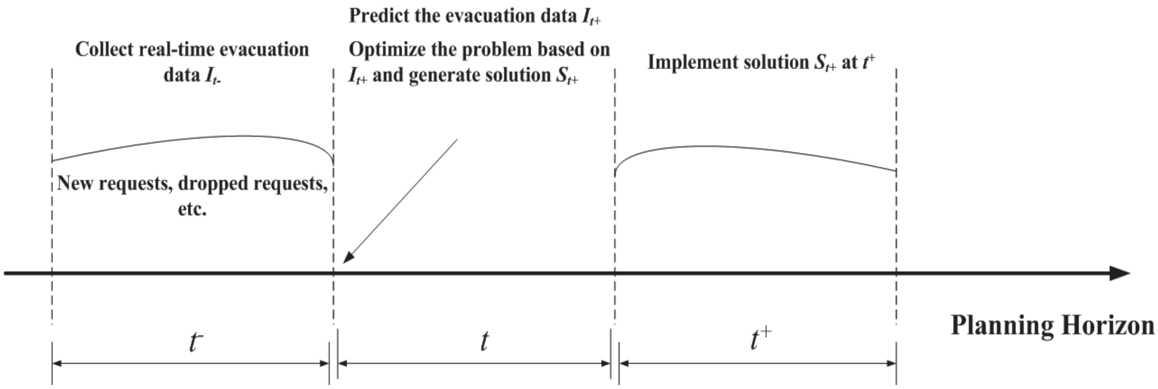

Predict the evacuation data I_{t+}
Optimize the problem based on I_{t+} and generate solution S_{t+}
Collect real-time evacuation data I_{t-}
Implement solution S_{t+} at t^+
New requests, dropped requests, etc.
Planning Horizon
t^-
t
t^+

Figure 4 Time Intervals over the Planning Horizon

Notations applied to the CDVRPPD model are as follows:

N_t = Set of unfulfilled pickup points in *t*, $\forall t \in T$.

F_t = Set of fulfilled pickup points in *t*, $\forall t \in T$.

E_t = Set of new pickup points in *t*, $\forall t \in T$.

V_t = Set of vertices in *t*, $V_t = N_t \bigcup S \bigcup M \bigcup L_t, \forall t \in T$.

A_t = Set of arcs in *t*, $\forall t \in T$.

c_{ijt} = Travel time on arc (i, j) in *t*, $\forall (i, j) \in A_t, \forall t \in T$.

c_{ijt}' = Predicted travel time on arc (i, j) in *t*, $\forall (i, j) \in A_t, \forall t \in T$.

c_{rt} = Travel time of a feasible route *r* in *t*, $\forall r \in R_t$, $\forall t \in T$.

H = Maximum allowed evacuation time.

l = Interval length.

T = Set of intervals, $T = \{t_0, t_1, \ldots, t_{\lfloor H/l \rfloor}\}$.

t = Time interval, $t \in T$.

t^- = Preceding interval to *t*, $\forall t \in T / t_0$.

t^+ = Subsequent interval to *t*, $\forall t \in T / t_{\lfloor H/l \rfloor}$.

R_t = Set of feasible routes in t, $\forall t \in T$.

$R_t{}'$ = Subset of feasible routes in t, $\forall t \in T$.

L_t = Set of vehicle locations in t, $\forall t \in T$.

c_{ijt} = Cost of using arc *(i, j)* in t, $\forall t \in T$, where $c_{ijt} = c_{ijt} - \frac{\delta_{it}}{2} - \frac{\delta_{jt}}{2}$

u_{it} = Vehicle load after visiting pickup point i in t, $\forall i \in N_{t^+} \bigcup L_{t^+}$, $\forall t \in T$.

As shown in Figure 5, at the beginning of interval *t*, all the evacuation data, including vehicle locations L_t, unfulfilled demands N_t, and arc travel times c_{ijt}, are updated. A CDVRPPD model is then formulated for generating transit vehicle routes applicable in t^+. The CDVRPPD model is proactive that the evacuation data in t^+ are estimated based on the evacuation data in t^-.

Link travel time is predicted by weighted moving average (Hunter, 1986). The weighted moving average method uses a weighting factor which gives more importance to recent observations while not discarding the older observations. The predicted travel time on arc (*i*, *j*) in t^+, $c^{'}_{ijt^+}$, is calculated using Equation (3.45) – (3.46).

$$c^{'}_{ijt^+} = \lambda c_{ijt} + \left(1-\lambda\right)\tau_{ijt^-} \tag{3.45}$$

Where, c_{ijt} is the observed travel time on arc (*i*, *j*) in *t*. λ is the weighting factor that $0 < \lambda \le 1$. τ_{ijt^-} is the average of observed travel times on arc (*i*, *j*) in t^-, which is calculated using Equation (3.46).

$$\tau_{ijt^-} = \frac{\sum_{t=t_0}^{t^-} c_{ijt}}{h} \tag{3.46}$$

Where h is the number of observations recorded in the historical data set.

The robustness and accuracy of the weighted moving average method depends on the value of the weighting factor, λ. λ determines how responsive a forecast is to travel time surge. For a real-time system with short-term travel time forecasting, a value of $\lambda = 0.38$ is suggested by Raiyn and Toledo (2014). The major advantage of the weighted moving average method is that it minimizes the data storage and computing requirements, which makes it suitable for real-time applications.

The set of vehicle locations in t^+, L_{t^+}, is determined based on the vehicle locations in t, L_t, and the arc travel time c_{ijt}. L_{t^+} is constantly changed over time. It is necessary to include L_{t^+} when formulating the CDVRPPD model. Each vehicle k's location is usually considered as a depot where the vehicle k departs in t^+. Thus, the problem turns to be a CDVRPPD with $\left|L_{t^+}\right|+1$ depots, which consists of $\left|L_{t^+}\right|$ temporary depots at the vehicle locations and one real depot.

For the vehicle routing problem with multiple depots, one of the most common methods is clustering which assigns pickup points to a depot. This procedure is deemed as a Generalized Assignment Problem (GAP). Once the GAP is solved, the problem is decomposed into multiple single-depot problems. Vehicle-flow formulation and set partitioning formulation are two classical methods to model and solve the problem.

For the CDVRPPD with multiple depots, vehicle k's temporary location $L_{t^+}^k$ is counted as a depot; however, no vehicle other than vehicle k can start from L_t^k. In this particular case, the problem can be converted to a CDVRPPD with single depot by introducing dummy pickup points in the network. A dummy pickup point n_k is added at vehicle k's location. Demand of n_k equals to vehicle k's load, and service time at n_k is 0. Travel time from n_k to other nodes is calculated according to their distance. In particular, travel time from depot s_0 to n_k is set to 0, and travel times from pickup points and shelters to n_k are set to infinite. After adding $\left|L_{t^+}\right|$ dummy pickup points, all the temporary depots are replaced by dummy pickup points. The problem is reduced to a CDVRPPD with single depot. The cost of adding $\left|L_{t^+}\right|$ dummy pickup points is that $\left|L_{t^+}\right|$ rows are added into the model. However, the complexity of the model is greatly reduced.

The set of unfulfilled pickup points in *t*, N_t, is formulated by Equation (3.47).

$$N_t = \begin{cases} N, \text{ when } t = t_0 \\ N_{t^-} \bigcup E_{t^-} \setminus F_{t^-}, \text{ otherwise} \end{cases} \tag{3.47}$$

The set of unfulfilled pickup points in t^+, N_{t^+}, is predicted by Equation (3.48). When estimating N_{t^+}, it does not take into account the pickup requests that arrive in *t*. The pickup requests that arrive in *t* will be considered in the next interval. The total pickup points set in t^+, including both the real pickup points and the dummy pickup points, is $N_{t^+} \bigcup L_{t^+}$.

$$N_{t^+} = N_t \setminus F_t, \ \forall t, t^+ \in T \tag{3.48}$$

In order to explicitly describe the graph *G* in the dynamic environment, two sets of vertices are introduced when representing the network in *t*. Given a vertex *i*, $V_{it^+}^+$ is defined as the set of vertices *j* such that arc (*i, j*) is not prohibited in t^+, i.e., the vertices in the set of $V_{it^+}^+$ are directly reachable from *i*. Similarly, $V_{it^+}^-$ denotes the set of vertices *j* from which vertex *i* is directly reachable in t^+.

$$V_{it^+}^{+} = \begin{cases} N_{t^+} \bigcup L_{t^+}, \text{ if } i \in S \\ N_t \bigcup M \setminus \{i\}, \text{ if } i \in N_{t^+} \bigcup L_{t^+} \\ S, \text{ if } i \in M \end{cases}$$

$$V_{it^+}^{-} = \begin{cases} M, \text{ if } i \in S \\ S \bigcup N_{t^+} \setminus \{i\}, \text{ if } i \in N_{t^+} \\ S, \text{ if } i \in L_{t^+} \\ N_t \bigcup L_{t^+}, \text{ if } i \in M \end{cases}$$

Let γ_{irt^+} be a variable indicating if route *r* visits pickup point *i* in t^+.

$$\gamma_{irt^+} = \begin{cases} 1, \text{ if route } r \text{ visits pickup point } i \text{ in } t^+, \ \forall t^+ \in T, \ \forall r \in R_{t^+}', \ \forall i \in N_{t^+} \bigcup L_{t^+} \\ 0, \text{ otherwise} \end{cases}$$

The CDVRPPD model has a binary variable x_{rt^+} indicating whether route *r* is used in t^+.

$$x_{rt^+} = \begin{cases} 1, \text{ if route } r \text{ is used in } t^+, \ \forall t^+ \in T, \ \forall r \in R_{t^+}' \\ 0, \text{ otherwise} \end{cases}$$

For interval t^+, the master problem model of CDVRPPD is formulated by (3.49) – (3.53).

$$\min \sum_{r \in R_{t^+}'} c_{rt^+} x_{rt^+} \tag{3.49}$$

$$\sum_{r \in R_{t^+}'} \gamma_{irt^+} x_{rt^+} \geq 1, \ \forall i \in N_{t^+} \bigcup L_{t^+} \tag{3.50}$$

$$x_{rt^+} \in \{0,1\}, \ \forall r \in R_{t^+}', \ \forall t^+ \in T \tag{3.51}$$

$$R_{t^+}' \subseteq R_{t^+} \tag{3.52}$$

$$\forall t^+ \in T \tag{3.53}$$

The master problem model is to select transit vehicle routes over the planning horizon to fulfill pickup requests from registered evacuees and unregistered evacuees so as to minimize the total travel cost. Constraints (3.50) ensure that each pickup point *i* is covered by at least one route *r* in the routes set R_{t^+}. It also requires a sub-problem model to generate routes with negative reduced cost. Let $\delta_{t^+} = \left\{ \delta_{1t^+}, \delta_{2t^+}, \ldots, \delta_{\left| N_{t^+} \cup L_{t^+} \right|} \right\}$ be the set of dual variables associated with (3.50) and $\delta_{t^+} = \left\{ \delta_{1t^+}, \delta_{2t^+}, \ldots, \delta_{\left| N_{t^+} \cup L_{t^+} \right|} \right\}$ be the dual optimal solution. The sub-problem model is formulated as follows.

$$\min \sum_{i \in V_{t^+}} \sum_{j \in V_{t^+}} c_{ijt^+} z_{ijt^+} \tag{3.54}$$

$$\sum_{i \in M} \sum_{j \in S} z_{ijt^+} = 1 \tag{3.55}$$

$$\sum_{i \in V^-_{jt^+}} \sum_{j \in M} z_{ijt^+} = 1 \tag{3.56}$$

$$\sum_{i \in S} \sum_{j \in V^+_{it^+}} z_{ijt^+} = 1 \tag{3.57}$$

$$\sum_{i \in V_{t^+}} z_{iot^+} - \sum_{j \in V_{t^+}} z_{ojt^+} = 0,\ \forall o \in V_{t^+},\ o \neq i,\ o \neq j \tag{3.58}$$

$$z_{ijt^+} = 0,\ \forall i \in S,\ \forall j \in M \bigcup S \tag{3.59}$$

$$z_{ijt^+} = 0,\ \forall i \in N_{t^+} \bigcup L_{t^+},\ \forall j \in S \tag{3.60}$$

$$z_{ijt^+} = 0,\ \forall i \in M,\ \forall j \in N_{t^+} \bigcup L_{t^+} \bigcup M \tag{3.61}$$

$$u_{it^+} - u_{jt^+} + Q z_{ijt^+} \leq Q - d_j,\ \forall i, j \in N_{t^+} \bigcup L_{t^+},\ i \neq j,\ such\ that\ d_i + d_j \leq Q \tag{3.62}$$

$$d_i \leq u_{it^+} \leq Q,\ \forall i \in N_{t^+} \bigcup L_{t^+} \tag{3.63}$$

$$z_{ijt^+} \in \{0,1\},\ \forall i, j \in V_{t^+} \tag{3.64}$$

z_{ijt^+} is decision variable.

$$z_{ijt^+} = \begin{cases} 1, \text{ if arc } (i, j) \text{ is used in the shortest path in } t^+, \\ \forall t^+ \in T,\ \forall i \in V_{t^+},\ \forall j \in V_{t^+},\ i \neq j \\ 0, \text{ otherwise} \end{cases}$$

The CDVRPPD model is similar to the CVRPPD model. Both are formulated based on a master problem model and sub-problem model structure. Constraints (3.55) – (3.58) are flow conservation constraints. Constraints (3.59) – (3.61) are connectivity constraints. Constraints (3.62) – (3.63) are the sub-tour elimination constraint. Constraints (3.64) ensure the integrality.

Dynamic Interval

The interval length, *l*, in which the optimization process is performed, is directly related to the network size. It is an important parameter in the CDVRPPD model development. When a new pickup request is collected in t^-, it will be processed in *t*, and then an updated routing plan will be implemented in t^+. Hence, a new pickup request has to wait at least one interval until an updated routing plan is implemented. On one hand, a short interval is beneficial to decreasing the waiting time of the new pickup request; on the other hand, a long interval is imperative at the initial stage of the planning horizon due to the computational burden. In instances where new pickup requests from unregistered evacuees are infrequent, the network size will decrease after the first several intervals. As a result, the computational burden will be reduced. For the CDVRPPD, it is necessary to adjust the interval length dynamically in order to keep the model reacting to the evacuation data updates.

In order to overcome the deficiency of fixed-length interval, the dynamic interval is implemented. The length of interval t is calculated based on the computation time in t^{-}.

$$l_t = t_{ct^-}\beta,\ t, t^- \in T,\ \beta > 1 \tag{3.65}$$

Where, l_t is the length of interval t. t_{ct^-} is the computation time in t^{-}. β is the incremental factor which represents the percent of increase. At the initial stage of the evacuation process, the network size increases with the new pickup request coming into the system. In response, the interval length will increase accordingly by multiplying the incremental factor β. It is expected that the computation time in t shows downtrend when the number of completed requests in t exceeds the number of new requests in t^{-}. In this case, the incremental factor β makes the interval length falling lag behind the computation time. It ensures a surplus of time each interval, which could be used to deal with uncertainties.

2-Stage Intermodal Evacuation Model

During an emergency event, transit may not be sufficient to manage the whole evacuation due to its limitations, such as limited bus capacity and driver availability. In this case, transit is usually integrated with other mass evacuation alternatives to ensure people are evacuated promptly. In this report, a 2-stage real time evacuation model is developed for the SmartEvac system. In the first stage, transit is used to transfer the evacuees to a train station. In the second stage, multiple trains are used to transfer the evacuees to safety shelters. The destination in the CDVRPPD model is revised from the shelters to the train station. In addition, multiple time windows are imposed on the bus arrival time at train station in order to make sure a smooth transition from transit to train transportation. The set of time windows W at the train station is defined as follows.

$$W = \left\{ w_i \mid w_i = \left(a_i, b_i \right) \right\} \tag{3.66}$$

Where, a_i and b_i are the start time and end time of time window w_i.

For interval t^{+}, the master problem model is modified based on the CDVRPPD master model formulated by (3.49) – (3.53). An extra constraint (3.67) is added to the model to ensure that the number of evacuees that arrive at the train station in a specific time window does not exceed the capacity of a train.

$$\sum_{\sigma_r \in w_i} u_{rt^+} \leq Q_{train}, \ \forall w_i \in W \tag{3.67}$$

Where, u_{rt+} is the route r's load when the related vehicle arrives the train station. σ_r is the route r's arrival time at the train station. Q_{train} is the train's capacity.

The sub-problem model (3.54) – (3.64) is modified by adding constraint (3.68) to ensure bus arrives at the train station in a specific time window within W.

$$\sigma_{t^+} \in w, \ w \in W \tag{3.68}$$

Solution Algorithm

In order to solve the CDVRPPD without enumerating all the routes, a column generation approach is applied to the problem. The general process of the column generation approach is presented as follows. First, an initial subset R_{t^+}' of all feasible routes R_{t^+} is enumerated. The LRMP, whose routes set is restricted to R_{t^+}', is then solved, and the dual solution is obtained. The dual solution is utilized in a sub-problem to determine if there are any routes that should be added to R_{t^+}' towards an optimum. The LRMP is then resolved with respect to the expanded R_{t^+}'. This process repeats until no additional routes can be found that further optimize the objective. At this point, the optimum solution to the LMP with R_{t^+} is found by solving the LRMP with R_{t^+}'. The optimum solution to the LMP is not necessarily an integer; the solution is actually fractional most of the time. If it is fractional, the final step is to solve the RMP as an integer problem in order to get an integer solution. A flow chart of the column generation method is shown in Figure 5.

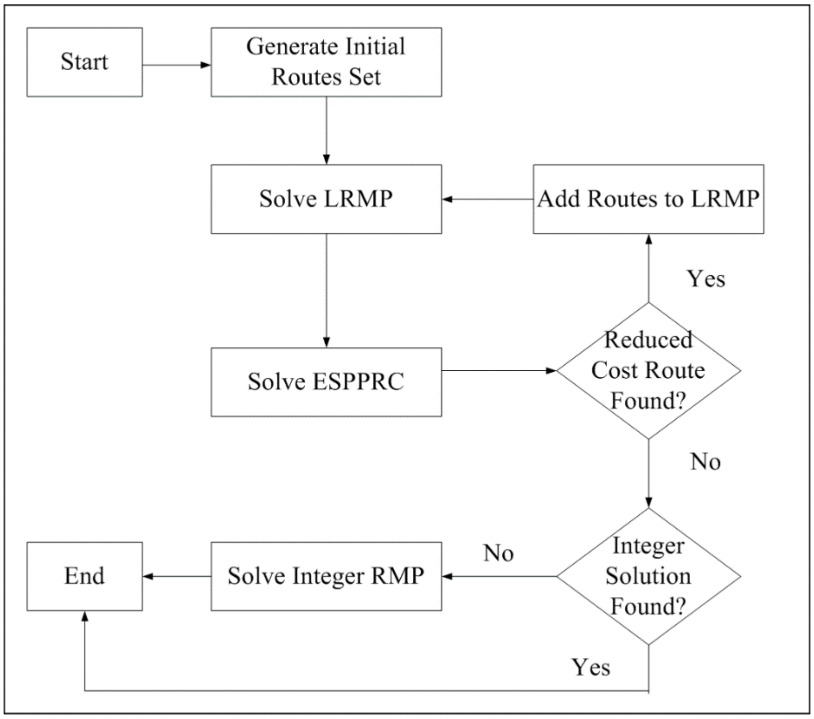


Figure 5 Column Generation Approach Flow Chart

The specific procedures of the column generation method are presented by the following steps:

Step 1. Create an initial subset of columns, R_{t^+}', $R_{t^+}' \subseteq R_{t^+}$.

Step 2. Solve the LRMP, and get the optimal solution y_{rt^+}, $r \in R_{t^+}'$ and the corresponding dual solution δ_{t^+}.

Step 3. Solve the ESPPCC sub-problem with δ_{t^+}. Identify routes *r*, $r \in R_{t^+}$ satisfying $c_{rt^+} < 0$.

Step 4. If $r \neq \varnothing$, add *r* into R_{t^+}' and go to step 2.

Step 5. If $r = \varnothing$, check if y_{rt^+} is an integer solution.

Step 6. If y_{rt^+} is integer, go to step 8.

Step 7. If y_{rt^+} is fractional, solve the integer RMP.

Step 8. End.

Initialization

Firstly, a set of columns is initialized for the LRMP. The initial set of columns needs to include at least a feasible solution to the LRMP. A common initial set is made of routes visiting a single pickup point, i.e. routes of type *C – N – M – C*. Since a good set of initial routes helps to generate routes with low reduced cost (Toth et al., 2001), quick heuristics are implemented to generate the initial routes set with high quality.

In *t₀*, the Clarke and Wright Savings Algorithm (Clarke and Wright, 1964) is applied to create initial routes. The Clarke and Wright Savings Algorithm is based on notion of savings. The basic idea is that a cost saving $s_{ij} = c_{i0} + c_{0j} - c_{ij}$ is generated when two routes $\left(0, \cdots, i, 0\right)$ and $\left(0, j, \cdots, 0\right)$ can be feasibly merged in to a single route $\left(0, \cdots, i, j, \cdots, 0\right)$. The specific procedures of the algorithm are implemented as follows.

Step 1. Create an initial routes set *R'* including |*N*| vehicle routes. Each route has the following route structure, $\left(0, i, m_i, 0\right),\ i \in N$, where m_i is the nearest shelter to pickup point *i*.

Step 2. Calculate the cost savings $s_{ij} = c_{im_i} + c_{m_i 0} + c_{0j} - c_{ij},\ \forall i, j \in N,\ i \neq j$, where $c_{m_i 0} = 0,\ \forall i \in N$. Rank the savings s_{ij} and list them in descending order. This creates the savings list.

Step 3. Process the savings list beginning with the topmost entry. For s_{ij}, find route $r_1,\ r_1 \in R'$ that starts with (*0*, *j*) and route $r_2,\ r_2 \in R'$ that ends with (*i*, m_i, *0*). Combine r_1 and r_2 into a new route r_3 by deleting (*0*, *j*) and (*i*, m_i, *0*) and introducing (*i*, *j*). If r_3 is feasible to the model, add r_3 into *R'* and remove r_1 and r_2 from *R'*.

Step 4. Iterate to the next entry in the savings list until the end.

The advantage of the Clarke and Wright Savings Algorithm lies in its simplicity and speed, which makes it suitable to generate a good set of initial routes. It typically runs within 0.5 seconds on Christofides, Mingozzi, and Toth's (1979) benchmark instances with 100 nodes.

The initialization step is handled differently in the interval $t^{+} \subseteq T \setminus \{t_0\}$. The initial routes set $R_{t^{+}}'$ in t^{+} is created based on the optimal routes set R_t in t. First, the Clarke and Wright Savings Algorithm is used to generate an initial routes set R_{ini} that serves the new pickup points in E_t^{-}. Second, the routes set R_t is updated. The vehicle routes that have been completed in t are removed from R_t. The pickup points that have been visited in t are removed from the routes as well. Third, an insertion algorithm is applied to R_t. For a new pickup point n in E_t and a route r in R_t, the algorithm inserts the new pickup point n to an arc (i, j) in r such that the incremental cost of inserting n between i and j is minimal. A flow chart of the insertion algorithm is shown in Figure 6.

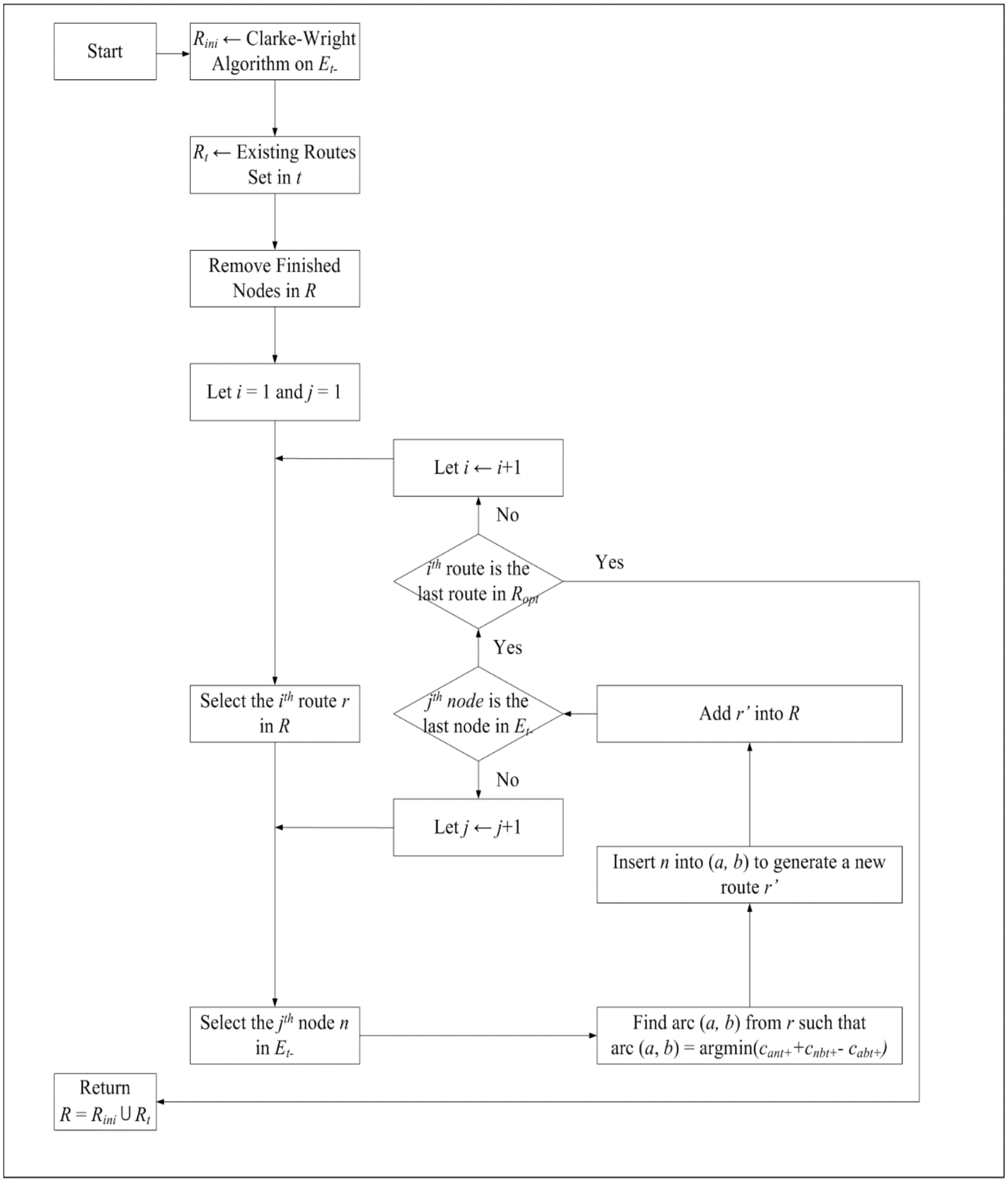

Start
R_{ini} ← Clarke-Wright Algorithm on E_{t-}
R_t ← Existing Routes Set in t
Remove Finished Nodes in R
Let i = 1 and j = 1
Let i ← i+1
No
i^{th} route is the last route in R_{opt}
Yes
Yes
Select the i^{th} route r in R
j^{th} node is the last node in E_{t-}
Add r' into R
No
Let j ← j+1
Insert n into (a, b) to generate a new route r'
Select the j^{th} node n in E_{t-}
Find arc (a, b) from r such that arc (a, b) = argmin(c_{ant+}+c_{nbt+}- c_{abt+})
Return $R = R_{ini} \cup R_t$

Figure ６ Insertion Algorithm Flow Chart

The specific procedures are described as follows:

For every pickup point *n* in E_t

For every route *r* in R_t^{opt}

Find an arc (*i*, *j*) in *r* such that $c_{int^+} + c_{njt^+} - c_{ijt^+}$ is minimal.

```
        $\arg\min_{(i,j)}\left(c_{int^+} + c_{njt^+} - c_{ijt^+}\right)$
        Construct a new route r’ by replacing (i, j) with (i, n, j).
        If r’ is feasible to the model, add r’ into $R^{opt}_{t}$.
    Update $R^{opt}_{t}$
End
```

Finally, the initial routes set is expanded by combining R^{opt}_{t} with R_{t^+}'.

2-**Cycle Elimination**

The objective of the pricing sub-problem is to identify the routes with negative reduced cost. The first step is to find the shortest path to each pickup point. This step is considered to be |*N*| ESPPCCs, each of which is NP-hard (Dror, 1994). For each ESPPCC, the task is to find the shortest partial path *r* from Node 0 to Node i, $i \in N$. Since shelters are not involved in finding the shortest partial paths, the network can be simplified by removing the shelters. Solving the ESPPCC is the most time consuming procedure in the column generation and thus significantly affects the performance of the optimization. Algorithms solving the ESPPCC in the literature include dynamic programming, branch-cut, and classic heuristics.

In this section, a cycle elimination (CE) algorithm is proposed based on standard labeling techniques presented by Desrochers (1988), Beasley and Christofides (1989), and Feillet et al. (2004). The CE algorithm first turns the ESPPCC to 2-cycle SPPCC by allowing cycles with length ≤ 2. Then a resource constraint is iteratively imposed upon the model to eliminate cycles with length > 2. Resource in the ESPPCC is related to capacity, time, and node availability etc., whose consumption is always nonnegative. The fundamental of the CE algorithm is based on Desrochers’ (1988) labeling algorithm which associates each potential partial path with a label indicating the consumption of resources.

The CE algorithm creates labels for each node $i,\ i \in N$. Each label l_i represents a partial path X_i from node 0 to node i. l_i includes a pointer $Pre(l_i)$ which links to l_i's parent label. l_i's parent label is defined as the label from which l_i is generated. Let $q(l_i)$ denote the capacity consumed on path X_i and $c(l_i)$ denote the travel cost associated with the path X_i. Thus a label l_i is represented as $l_i(Pre, q, c)$. The algorithm repeatedly extends each label to its successors until all labels have been extended in all feasible ways. The extension is operated by appending an arc (i, j) to path X_i to generate a new path X_j. When a label l_i is extended to a label l_j, the capacity consumption and the path cost are updated as follows,

$$q\left(l_j\right) = q\left(l_i\right) + d_j \tag{3.69}$$

$$c\left(l_j\right) = c\left(l_i\right) + c_{ij} \tag{3.70}$$

A new label $l_j(pre, q, c)$ is generated only if,

$$q\left(l_j\right) < Q \tag{3.71}$$

It is noted that (3.69) is strictly non-decreasing since $d_j > 0$ for all $j \in N$. The extension of a label l_i is denoted by $Ext(l_i)$.

Dominance Rule

The efficiency of the CE algorithm highly depends on the number of labels generated. Since the extension operation creates exponential number of labels, it is necessary to discard the labels that will not lead to an optimal solution. For this purpose, a dominance rule is applied in the label extension so that the algorithm records only non-dominated labels.

If there are two labels $l_{i(1)}$ and $l_{i(2)}$ associated with node i satisfying $q(l_{i(1)}) \leq q(l_{i(2)})$, $c(l_{i(1)}) \leq c(l_{i(2)})$, and $l_{i(1)} \neq l_{i(2)}$, then any feasible extension from label $l_{i(2)}$ will be also feasible from label $l_{i(1)}$. In addition, new labels created based on label $l_{i(1)}$ will always be better than the labels created based on label $l_{i(2)}$, in terms of travel cost (if the objective is to minimize travel cost). Therefore, the label $l_{i(2)}$ can be discarded. The dominance rule is defined that $l_{i(1)}$ dominates $l_{i(2)}$, denoted by $l_{i(1)} \prec_{dom} l_{i(2)}$, if and only if the following conditions are met.

$$q\left(l_{i(1)}\right) \leq q\left(l_{i(2)}\right) \tag{3.72}$$

$$c\left(l_{i(1)}\right) \leq c\left(l_{i(2)}\right) \tag{3.73}$$

$$l_{i(1)} \neq l_{i(2)} \tag{3.74}$$

Figure 7 illustrates the dominance rule that any label in the shaded area will dominate label $l_{i(2)}$.

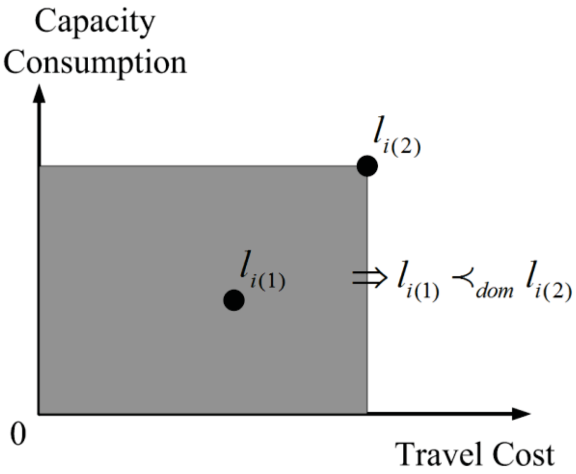


Figure 7 Illustration of Dominance Rule

After a new label l_i is generated, it is necessary to check whether the new label is dominated by other labels associated with the same node, and whether the new label dominates other labels. The procedure of dominance check to l_i is denoted by $Dom(l_i)$.

Any label which has been identified as being dominated by other labels will be discarded because any extension from the dominated label will be worse than the extension from the dominant label.

Enhanced Dominance Rules for 2-Cycle Elimination

The above dominance rule is applicable in the context of finding a non-elementary shortest path. Because of the existence of negative cost arcs, the relaxation of elementary constraint results in a lot of paths with cycles. This typically weakens the lower bound which leads to a bigger branch-and-bound tree. To improve the lower bound, Houck et al. (1980) proposed an algorithm for solving the SPPRC with 2-cycle elimination. Larsen (1999) extended Houck's algorithm with new definition of labels. In this section, Larsen's method is enhanced by improving the dominance rules.

Let (i, q) denote a state of node i, which indicates the capacity consumption. For each state, the algorithm generates two types of labels as follows. A new parameter *Typ* is appended to l_i. *Typ*(l_j) denotes the type of l_i.

1. Strong-dominant label that *Typ*(l_j) = *Strong*. A strong-dominant label is the prevailing label that dominates the extension. However, a strong-dominant label l_i cannot be extended to its predecessor node. Let v(l_i) denote the associated node of l_i. l_i 's predecessor node is the node which l_i's parent label is associated with, denoted by v(*Pre*(l_i)).
2. Weak-dominant label that *Typ*(l_j) = *Weak*. A weak-dominant label is dominated by the strong-dominant label. A weak-dominant label has the potential of being extended to the strong-dominant label's predecessor node. It actually provides an alternative path when the extension of the strong-dominant label forms a 2-cycle.

The algorithm can effectively eliminate 2-cycle by introducing a weak-dominant label for each state. As a result, the total number of labels is doubled. Therefore, the computational complexity remains the same.

Strong-dominant label and weak-dominant label have different extension rules. When a label l_i is extended to generate a label l_j, the following extension rules are applied.

1. If $Typ(l_i) = Strong$, (3.69) – (3.71) are applied. l_i is not permitted to extend to $v(l_j)$ if $v(pre(l_i)) = v(l_j)$. When $v(pre(l_i)) = v(l_j)$, the weak-dominant label is extended to $v(l_j)$ instead of l_i and (3.69) – (3.71) are applied.
2. If $Typ(l_i) = Weak$, l_i is extendable on the condition that $v(l_j)$ is the predecessor node of the strong-dominant label which dominates l_i, otherwise, l_i is not extendable. When $v(l_j)$ is the predecessor node of the strong-dominant label, l_i is extended instead of the strong-dominant label and (3.69) – (3.71) are applied.

In summary, a strong-dominant label is extendable to any node except its predecessor node. A weak-dominant label is not extendable to any nodes other than the predecessor node of the strong-dominant label. In addition, any extension has to satisfy (3.69) – (3.71).

New dominance rules are added in addition to (3.72) – (3.74), which are described as follows. Assume that $l_{i(1)}$ is an old label at node i and $l_{i(2)}$ is a new generated label at node i. If $l_{i(1)} \prec_{dom} l_{i(2)}$ according to (3.72) – (3.74), $l_{i(2)}$ can be discarded only if one of the following conditions are satisfied.

1. $Typ(l_{i(1)})$ = Strong and $v(Pre(l_{i(1)})) = v(Pre(l_{i(2)}))$.
2. $Typ(l_{i(1)})$ = Weak.

When $Typ(l_{i(1)})$ = Strong and $v(Pre(l_{i(1)})) \neq v(Pre(l_{i(2)}))$, the new generated label $l_{i(2)}$ will proceed to compare with the weak-dominant label dominated by $l_{i(1)}$, to determine whether it can replace the weak-dominant label.

If $l_{i(2)} \prec_{dom} l_{i(1)}$, then $l_{i(1)}$ can be discarded only if one of the following conditions are satisfied.

1. $Typ(l_{i(1)})$ = Strong and $v(Pre(l_{i(1)})) = v(Pre(l_{i(2)}))$.
2. $Typ(l_{i(1)})$ = Weak.

Similarly, when $Typ(l_{i(1)})$ = Strong and $v(Pre(l_{i(1)})) \neq v(Pre(l_{i(2)}))$, the old label $l_{i(1)}$ will become the weak-dominant label which is dominated by $l_{i(2)}$.

The pseudo code of the algorithm is presented as follows. $\mathit{\Gamma}$ represents the set of labels which have not been extended. Only strong-dominant labels are placed in $\mathit{\Gamma}$. Labels in $\mathit{\Gamma}$ are placed in lexicographical order. Given two labels $l_i(Pre, q, c, Typ)$ and $l_j(Pre, q, c, Typ)$, l_i is lexicographically smaller than l_j if $q(l_i) < q(l_j)$. $Ext(l_i)$ is the extension procedure which extends label l_i to its successors. The capacity constraint is checked and only feasible labels are produced. $Dom(l_i)$ is the procedure which applies the dominance rule to the new generated label. When a new label $l_i(Pre, q, c, Typ)$ is generated at node i, the dominance rule is applied to check whether the new label is dominated by the old label associated with state $(i, q(l_i))$. Then the strong-dominant label and the weak-dominant label associated with state $(i, q(l_i))$ are updated according to the results of the dominance check. The specific procedures of the 2-cycle elimination algorithm are presented as follows.

Step 1. Initialization
Initialize the label $l_0 = (Null, 0, 0, Strong)$ for node 0. Initialize $l_i = (Null, q, +\infty, Strong)$ and $l_i' = (Null, q, +\infty, Weak)$ for all other node i, $i \in N$ and q, $0 < q \leq Q$. Then, let $\mathit{\Gamma} = \{l_0\}$.
Step 2. Label Selection
If $\mathit{\Gamma} = \emptyset$, go to Step 4.
Else, select the first label l_i in $\mathit{\Gamma}$. Then Remove l_i from $\mathit{\Gamma}$.
Step 3. Label Extension
For all $(i, j) \in A$, $j \neq 0$
Create a new label $l_j \leftarrow Ext(l_i)$.
Apply the dominance rule to l_j, $Dom(l_j)$.

Step 4. Insert all new generated strong-dominant labels into Γ in lexicographical order. If $\Gamma \neq \emptyset$, go to Step 2.

Step 5. Stop. All labels are extended in all feasible ways.

The algorithm generates a set of strong-dominant labels at each node i, $i \in N$. Then the best label at node i, which indicates a shortest path from node 0 to node i, is found.

Computational Results

Benchmark instances available at http://goo.gl/9tclrK are used to evaluate the proposed algorithms. The instances were performed on an Intel P8200 Duo 2.2 GHz PC with 4G memory. CPLEX was used as the LP and MIP solver. For each instance, the lower bound, the number of columns in R_t, and the total computational time taken in CPU seconds were reported. The results from Agarwal, Mathur, and Salkin (1989), Bixby (1998), and Hadjiconstantinou, Christofides, and Mingozzi (1995) were also presented in Table 1– Table 4 in comparison.

Table 1 Computational Results

No.	Instance	Nodes	Z^*	Z^{LB}	Effectiveness of Z^{LB}	Cols	Time for Generating Columns (s)	Total CPU Time (s)
1	E016-03m	15	273	270	98.9%	264	1.4	3.9
2	E021-04m	20	353	353	100.0%	492	1.1	3.5
3	E026-08m	25	607	606	99.8%	642	1.0	2.4
4	E031-09h	30	610	605	99.2%	1137	7.5	19.1
5	E036-11h	35	698	698	100.0%	1644	6.5	13.7
6	E041-14H	40	859	859	100.0%	1829	21.0	59.5
7	E051-05e	50	521	518	99.4%	4904	53.1	138.8
8	E076-10e	75	830	815	98.2%	8919	126.5	335.1
9	E101-08e	100	815	804	98.6%	10248	744.8	2381.2
10	E101-10c	100	820	803	97.9%	14346	801.2	2503.2

Table 2 Results Comparison of Agarwal, Mathur, and Salkin (AMS) and CE

Problem	*n*	*Z**	AMS		CE Algorithm		
			Z^{LB}	Effectiveness of Z^{LB}	Z^{LB}	Effectiveness of Z^{LB}	Total CPU Sec
E016-03M	16	273	268	98.2%	270	98.9%	3.9
E021-04M	21	353	351	99.4%	353	100.0%	3.5
E022-04G	22	375	374	99.7%	369	98.4%	1.1
E026-08M	26	607	606	99.8%	606	99.8%	2.4

Table 3 Results Comparison of Bixby and CE

Problem	*n*	*Z**	Bixby		CE Algorithm		
			Z^{LB}	Effectiveness of Z^{LB}	Z^{LB}	Effectiveness of Z^{LB}	Total CPU Sec
E023-03G	23	568	566	99.6%	567	99.8%	23.5
E030-04S	30	503	503	100.0%	503	100.0%	9.5

Table 4 Results Comparison of Hadjiconstantinou, Christofides, and Mingozzi (HCM) and CE

Problem	*n*	*Z**	HCM		CE Algorithm		
			Z^{LB}	Effectiveness of Z^{LB}	Z^{LB}	Effectiveness of Z^{LB}	Total CPU Sec
E036-llH	36	698	694	99.4%	698	100.0%	13.7
E041-14H	41	859	852	99.2%	859	100.0%	59.5
E051-05E	51	521	516	99.0%	518	99.4%	138.8
E076-10E	76	830	815	98.2%	815	98.2%	335.1
E101-08E	101	815	792	97.2%	804	98.7%	2381.2

In Table 1– Table 3, the name of the instance and the number of nodes involved are listed. The value of the optimal integer solution Z^*, lower bound Z^{LB}, and the effectiveness of

Z^{LB}, which is the difference between the optimum and lower bound, are provided for the AMS, Bixby, HCM methods, and CE algorithm, respectively. In addition, the computational time of the CE algorithm is provided in the table, however, no data is found regarding the computational time of the AMS, Bixby, HCM methods. Among nine of the eleven instances, the lower bound Z^{LB} provided by the CE algorithm proposed is tighter than the other three methods. There is only one out of eleven case that the lower bound Z^{LB} provided by the CE algorithm is slightly not as good as that generated by AMS (a difference of 0.7%).

SIMULATION AND CASE STUDY

In this chapter, a case study of the Hurricane Gustavo evacuation in Gulfport is proposed to evaluate the SmartEvac system. The case study is based on CORSIM simulation, which provides dynamic travel time for the system. Scenarios corresponding to different evacuation situations are built in the simulation. The capability of the SmartEvac system working in a dynamic environment is validated by the case study.

CORSIM Network Development

In this case study, emergency evacuation scenarios are replicated based on the data from the Hurricane Gustavo emergency evacuation in 2008. There are 182 registered evacuees across 66 pickup points in the Mississippi Gulf Coast region. In addition, based on CTA's experience, 46 unregistered evacuees across 30 pickup points are considered in this case study. The unregistered evacuees are expected to call for help at any time during the emergency evacuation. Three shelters in the region provide temporary housing for the evacuees. The distribution of shelters and pickup points with registered evacuees is shown in Figure 8.

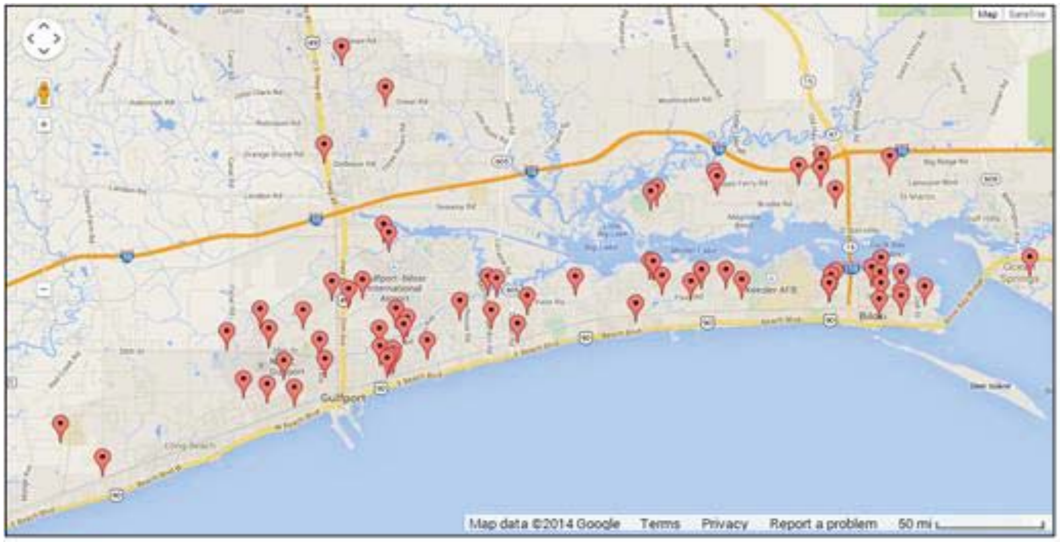


Figure 8 Distribution of Shelters and Pickup Points with Registered Evacuees

Detail information about shelters and pickup points with registered evacuees will be available upon request, which is subject to CTA's approval.

A homogenous fleet of transit vehicle is used in the emergency evacuation. The capacity of the transit vehicle is 30. Each transit vehicle has onboard equipment that is able to receive orders from the SmartEvac system in real time. The dwell time at each pickup point is two minutes.

According to the CTA's evacuation plan, the emergency evacuation started at the 7:00 AM rush hour. It is assumed that calls from the unregistered evacuees will evenly arrive with 3-minute interval.

CORSIM Simulation Development

The transportation network data and evacuation data used to build the CORSIM network are collected from field survey, CTA, and the Office of Engineering, etc. Figure 9 shows the data used in the simulation.

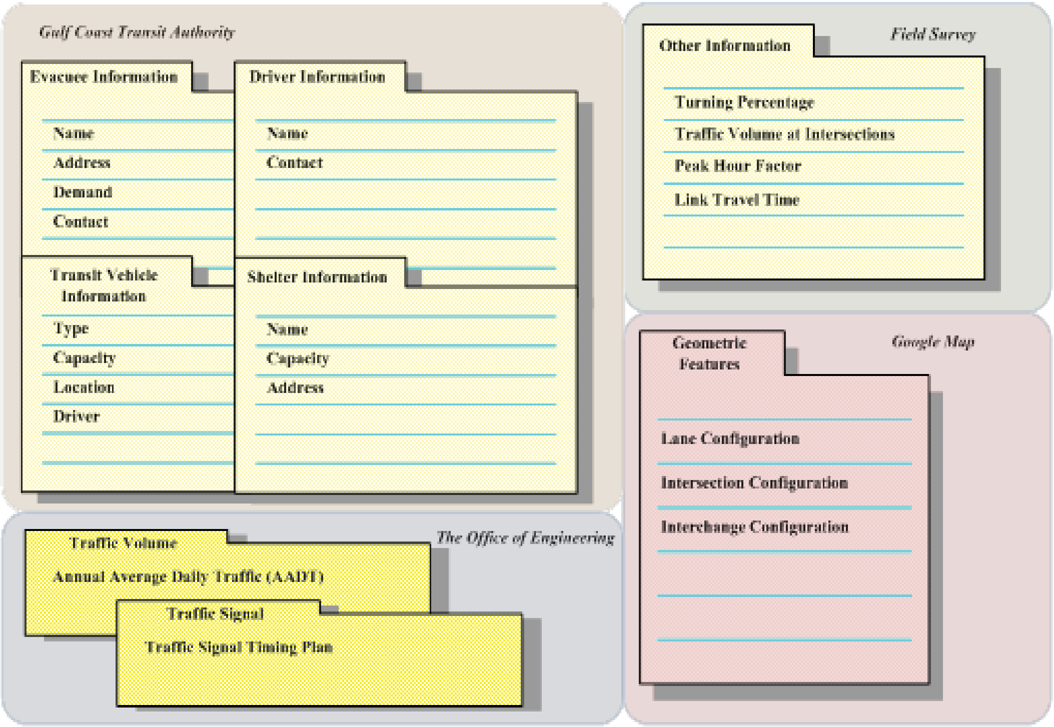


Figure ９Data Used in the CORSIM Simulation

Field surveys were conducted at 23 major intersections in the Gulfport Coast region. These intersections are mainly distributed along Pass Road, Highway 605, Canal Road, and Popps Ferry Road. Radar detectors and manual counters were deployed at the 23 intersections for five days to collect daily traffic volumes, peak hour traffic volumes, and turning percentage data. Turn prohibitions are implemented at specific intersections where prohibitory traffic signs are placed.

The CORSIM network is shown in Figure １０. Interstate 10 runs east and west of the Mississippi Gulf Coast region. Other major roadways include I-110, U.S.90, U.S.49, Pass Road, Highway 605, and Highway 67. The CORSIM network consists of 1,632 links and 1,341 nodes, in which 146 nodes are signalized intersections. The traffic signal timing plans were extracted from the City Engineering ACTRA system.

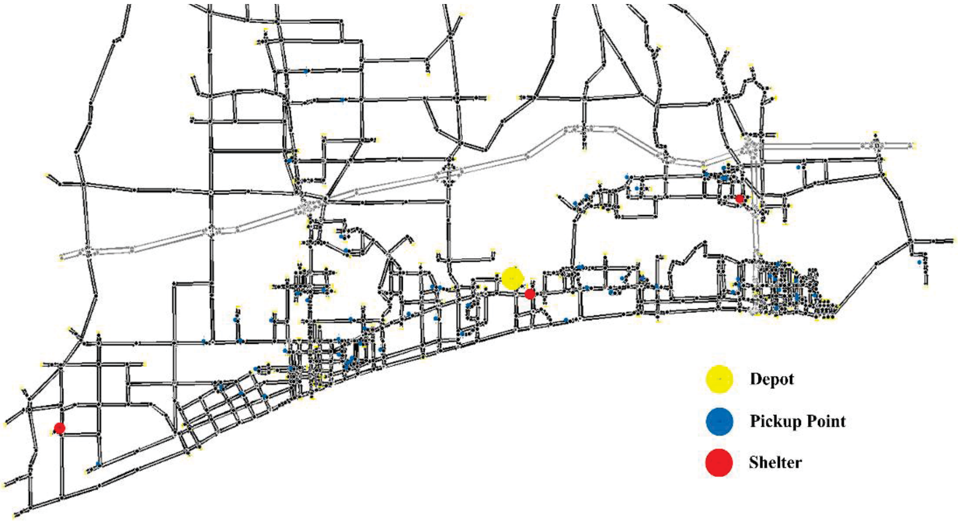


Figure １０ CORSIM Network of Gulfport Region

In addition to the intersections and transition nodes, depots, shelters, and pickup points are coded in the CORSIM network. In Figure １０, depots, shelters, and pickup points with registered evacuees are marked with yellow, red, and blue color in the CORSIM network, respectively. The shortest travel times among vertices including depots, shelters, and pickup points are calculated using a modified Dijkstra Algorithm (Wen, 2012) in which turn prohibitions are considered.

The length of the simulation is two hours which is consistent with the CTA's evacuation plan. Thirty intervals, $t_1, t_2, \cdots, t_{30}$, with equal length of three minutes are implemented in the simulation.

The CORSIM simulation model is fine-tuned with morning rush hour travel time data collected in the field. The simulated travel times on two major roads in the area, a U.S. 90 segment between the Bay St. Louis Bridge and the Biloxi Bay Bridge, and a Pass Road segment between U.S. 49 and Rodeo Drive, compared with those from Google Maps and historical data of 2008, are shown in Figure １１.

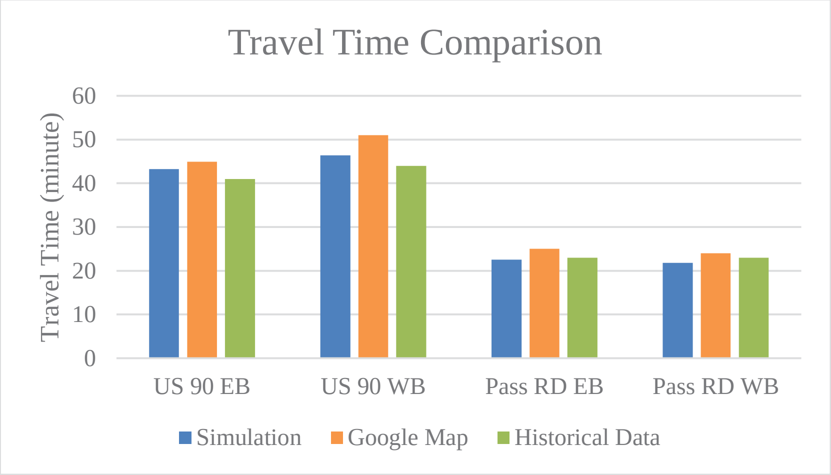


Figure １１Travel Time Comparison between Simulation, Google Map, and Historical Data

Results of Case Study

To evaluate the performance of the SmartEvac system in an emergency evacuation, especially when dynamic factors, such as unregistered evacuees' pickup requests and network interruptions, are considered, the following emergency evacuation scenarios are developed.

Scenario 1

Scenario 1 is developed as a base scenario. There are no dynamic factors in the emergency evacuation which means that the transit vehicle routes remain fixed all the time.

Results from the SmartEvac system are displayed in Figure １２. There are seven transit vehicles used in the emergency evacuation. The total travel time of all the seven transit vehicle routes is 417.9 minutes. The total computation time is 157 seconds while the time for generating columns is 68 seconds.

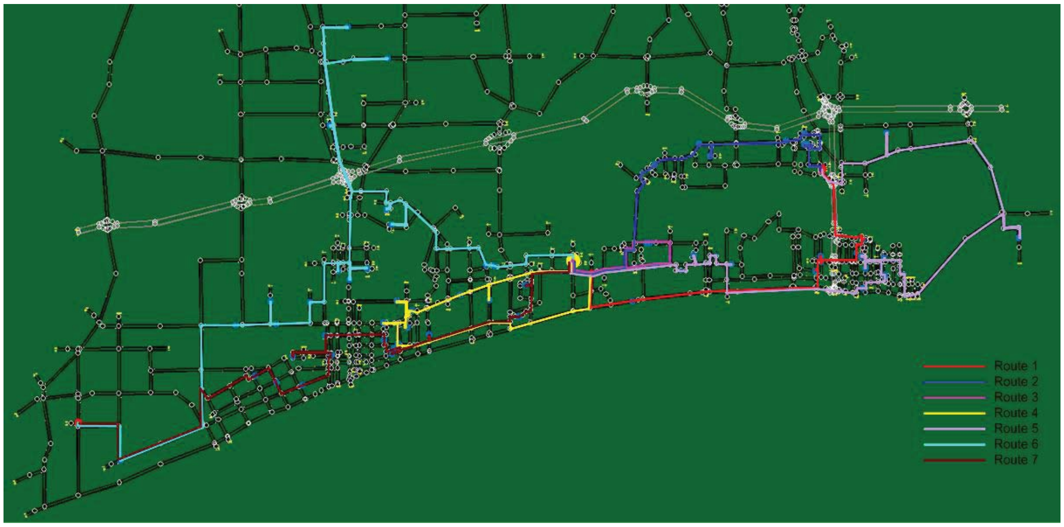


Figure １２ Results from the Scenario 1

Each individual transit vehicle route is listed as follows.

Route 1: cost = 40.5 minutes and load = 30.

Node 0 - Node 27 - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 1

Route 2: cost = 43.0 minutes and load = 16

Node 0 - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - Node 62 - Node 1

Route 3: cost = 20.4 minutes and load = 18

Node 0 - Node 24 - Node 20 - Node 23 - Node 3

Route 4: cost = 48.4 minutes and load = 29

Node 0 - Node 59 - Node 37 - Node 48 - Node 32 - Node 68 - Node 36 - Node 47 - Node 3

Route 5: cost = 79.9 minutes and load = 30

Node 0 - Node 10 - Node 13 - Node 5 - Node 12 - Node 18 - Node 8 - Node 14 - Node 15 - Node 19 - Node 22 - Node 7 - Node 66 - Node 25 - Node 1

Route 6: cost = 114.1 minutes and load = 29

Node 0 - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - Node 34 - Node 49 - Node 33 - Node 55 - Node 42 - Node 53 - Node 51 - Node 65 - Node 64 - Node 2

Route 7: cost = 71.7 minutes and load = 30

Node 0 - Node 41 - Node 31 - Node 35 - Node 40 - Node 52 - Node 30 - Node 46 - Node 58 - Node 45 - Node 43 - Node 54 - Node 56 - Node 38 - Node 2

Scenario 2

In order to replicate the scenario that unregistered evacuees call for pickup after the emergency evacuation starts, Scenario 2 is created based on Scenario 1, but new pickup requests from unregistered evacuees are generated per interval.

Take interval t_1 as an example. In t_1, a new pickup request at Node 70 with demand of 1 is added in the system. In response to the new request, the SmartEvac system re-optimizes the transit vehicle routes in t_2. After optimization, three out of the seven routes are adjusted. The updated transit vehicle routes will be implemented in t_3, as shown in Figure １３.

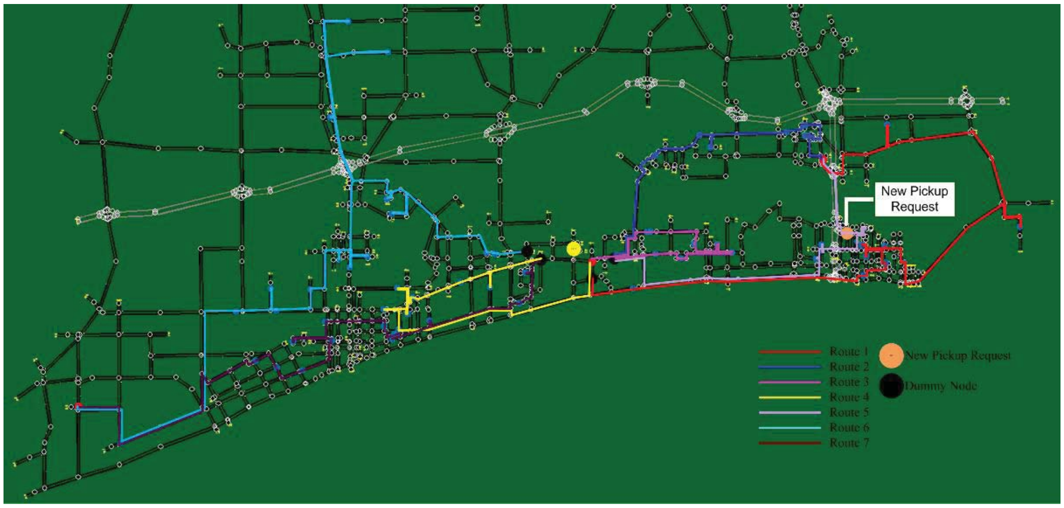


Figure １３ Transit Vehicle Routes after Re-optimization in Scenario 2 t1

The total cost of the re-optimized vehicle routes is 398.7 minutes. There are still seven transit vehicles used in the emergency evacuation. The computation time for the re-optimization is 173 seconds. In comparison with the transit vehicle routes in Scenario 1, Route 1, Route 3, and Route 5 are re-optimized due to the new pickup request at Node 70. The revisions are shown as follows.

Route 1: cost = 65.6 minutes and load = 21

Dummy Node - Node 27 - Node 18 - Node 8 - Node 14 - Node 15 - Node 19 - Node 22 - Node 7 - Node 66 - Node 25 - Node 1

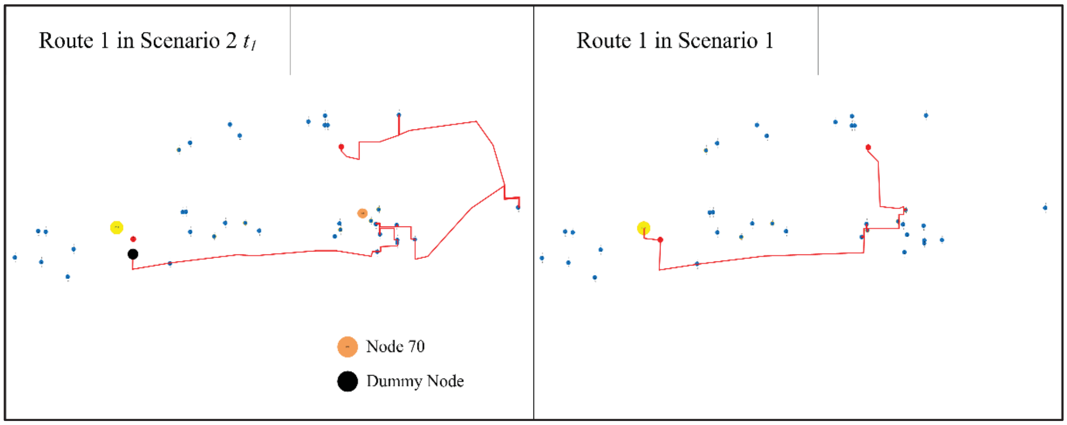


Figure １４Comparison of Route 1 between Scenario 1 and Scenario 2 t1

Route 3: cost = 33.4 minutes and load = 28

Dummy Node - Node 23 - Node 10 - Node 12 - Node 5 - Node 13 - Node 20 - Node 24 - Node 3

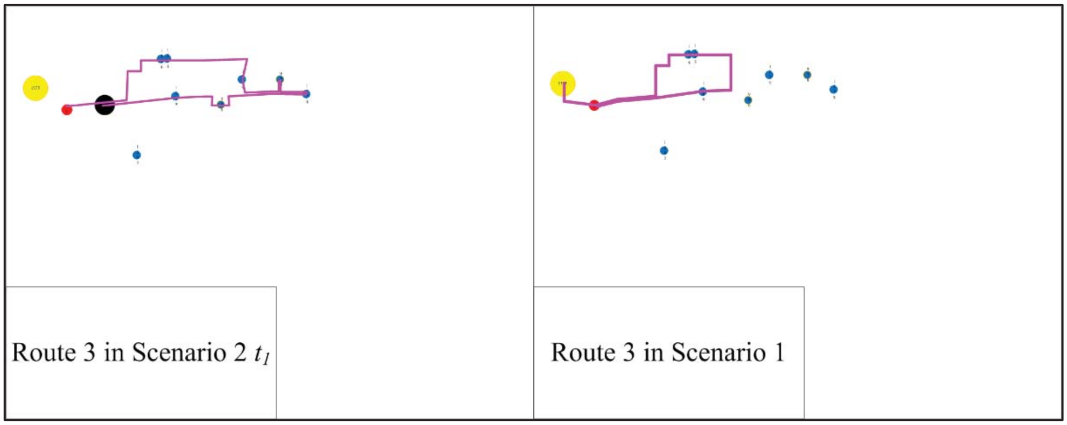


Figure １５ Comparison of Route 3 between Scenario 1 and Scenario 2 t1

Route 5: cost = 34.5 minutes and load = 30

Dummy Node - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 70 - Node 1

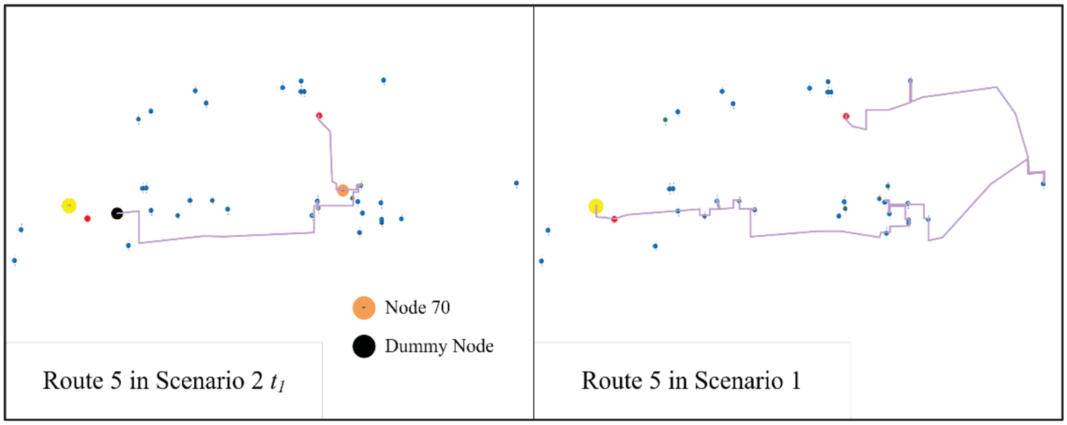


Figure １６ Comparison of Route 5 between Scenario 1 and Scenario 2 t1

The new added pickup request at Node 70 is serviced by Route 5. In order to explicitly explain how the SmartEvac system adjusts the vehicle routes for the new pickup request, all of the pickup points are distributed to eight zones based on their geographic location, as shown in Figure １７. The boundaries of the zones consist of major roads and bridges in the region, such as I-110, U.S.90, U.S.49, Pass Road, and Popps Ferry Bridge.

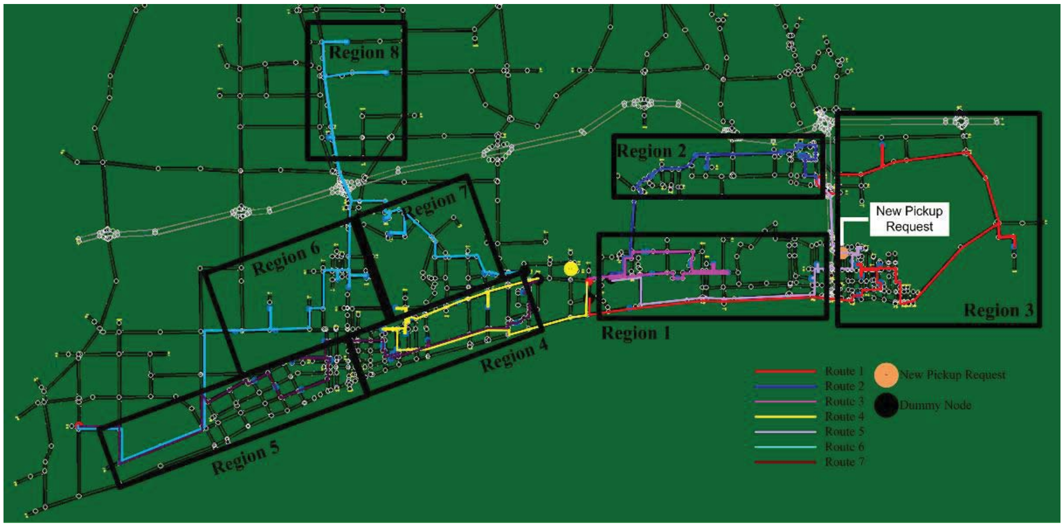


Figure １７ The Spatial Distribution of Zones of Pickup Points

The new pickup request, Node 70, is located in Zone 3. All of the pickup points in Zone 3 are serviced by two vehicles in Scenario 1: Vehicle 1 on Route 1 and Vehicle 5 on Route 5. Obviously, Vehicle 1 and Vehicle 5 are two candidates to pick up Node 70.

However, both Vehicle 1 and 5 are at full capacity, with no space left for node 70 according to the original routing plan in Scenario 1. Therefore, another vehicle is required to relieve Vehicle 1 or Vehicle 5's load in order to free up space for Node 70. As shown in Figure 4.5, there is only one vehicle route, Route 3 in Zone 1, which covers Route 5. No vehicle routes can cover Route 1 by making trivial revisions. Both Route 3 and Route 5 include the Pass Road section from Popps Ferry Road to Veterans Avenue. The SmartEvac system is able to reassign the pickup points in this section from Route 5 to Route 3 to release Vehicle 5's capacity. Now Vehicle 5 has sufficient capacity to pick up Node 70 because Node 10, Node 13, Node 5, and Node 12 are taken over by Vehicle 3.

However, Vehicle 5 is still not the best option to pick up Node 70 because Node 70 is farther away from Route 5 than Route 1. Therefore, the SmartEvac system swaps Vehicle 5's tasks with Vehicle 1's tasks. After this swap, Vehicle 1's tasks after picking up Node 27 are taken over by Vehicle 5, and all of Vehicle 5's tasks are taken over by Vehicle 1. Finally, Vehicle 5 turns to be the most appropriate vehicle to pick up Node 70 and Route 5 is revised by including Node 70.

A new pickup request per interval is received for 30 intervals. The SmartEvac system updates the pickup information and re-optimizes the transit vehicle routes accordingly. The results are summarized in Table 5.

Table 5 Results of Scenario 2 with Fixed Interval

Time Interval	Total Cost (Minute)	No. of Vehicle	Computation Time (Second)
t_0	417.9	7	157
t_1	398.7	7	171
t_2	377.3	7	168
t_3	345.3	7	140
t_4	317.4	7	121
t_5	294.6	7	48

t_{6}	278.0	7	31
t_{7}	236.4	6	12
t_{8}	219.6	6	10
t_{9}	192.5	5	3
t_{10}	177.0	5	2
t_{11}	175.9	5	4
t_{12}	157.0	5	2
t_{13}	154.4	5	2
t_{14}	147.7	5	2
t_{15}	143.7	5	3
t_{16}	130.5	4	6
t_{17}	113.6	3	1
t_{18}	115.2	3	1
t_{19}	100.3	3	1
t_{20}	89.3	3	1
t_{21}	93.3	3	1
t_{22}	98.5	3	1
t_{23}	110.6	3	1
t_{24}	101.8	2	1
t_{25}	92.1	2	1
t_{26}	97.0	2	1
t_{27}	91.8	2	1
t_{28}	93.2	2	1
t_{29}	110.1	3	1
t_{30}	118.7	3	1

Dynamic intervals could be implemented in the optimization process. The length of interval t_i is calculated based on the computation time in t_{i-1}. The initial time interval is

180 seconds, and the minimum time interval is 60 seconds. The incremental factor β is 110%. The results of Scenario 2 with dynamic intervals are listed in Table 6. Table 6 only shows the intervals in which the SmartEvac processing the new pickup requests.

Table 6 Figure 10 Results of Scenario 2 with Dynamic Interval

Total Cost (Minute)	Computation Time (Second)	Interval Length (Second)	Wait Time (Second)	New Request Arrival Time (Second)
417.9	156	180		90
399.7	171	172	262	285
376.3	165	188	255	472
345.2	137	182	249	613
320.8	119	151	259	790
296.5	51	131	213	995
289.9	34	60	68	1155
248.4	15	60	97	1345
231.6	11	60	87	1530
202.5	5	60	82	1680
187.0	5	60	112	1900
185.9	4	60	72	2083
167.0	3	60	69	2257
164.4	2	60	75	2433
157.7	6	60	79	2587
153.7	5	60	105	2801
138.5	5	60	71	2979
119.6	4	60	73	3122
121.2	3	60	110	3350
106.3	1	60	62	3518
95.3	1	60	74	3701
99.3	1	60	71	3891
104.5	1	60	61	4023
116.6	1	60	109	4225
105.8	1	60	87	4419
96.1	1	60	73	4581

101.0	1	60	91	4770
95.8	1	60	82	4931
97.2	1	60	101	5128
116.1	1	60	84	5331
124.7	1	60	61	

In Scenario 2, there is a total of 30 requests from unregistered evacuees added in the emergency evacuation. *Figure １８*shows the distribution of all evacuees, including both registered and unregistered evacuees.

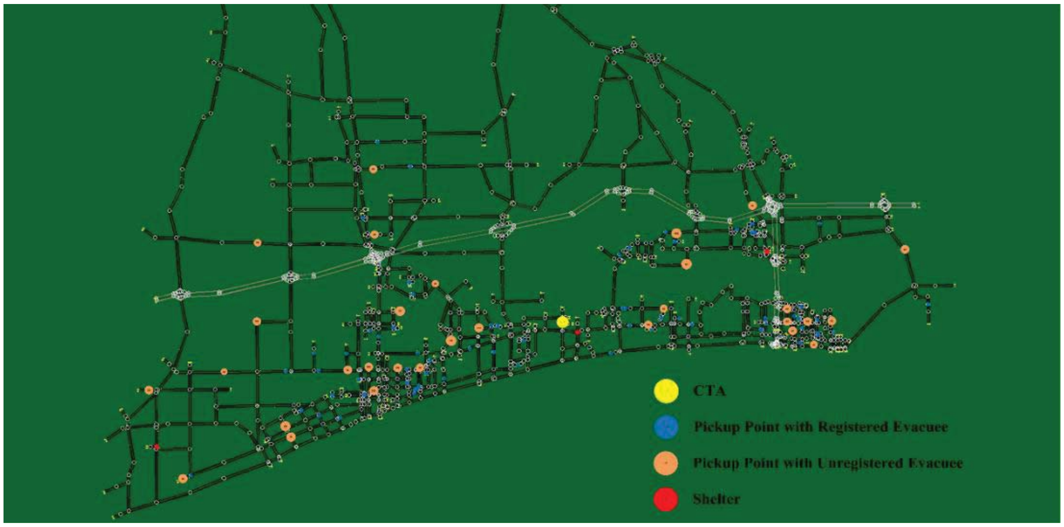

CTA
Pickup Point with Registered Evacuee
Pickup Point with Unregistered Evacuee
Shelter

Figure １８ CORSIM Network with Unregistered Evacuees

Scenario 3

Scenario 3 is developed based on Scenario 2 but certain incidents, such as traffic accidents and a broken bridge, are implemented.

Scenario 3(a)

Assuming that traffic accidents occur on U.S. 90 after the emergency evacuation starts, as shown in *Figure １９*, the travel speed on U.S. 90 is severely impacted by the accidents.

It is assumed that the average travel time on U.S. 90 in Scenario 3 is twice of what's in Scenario 2.

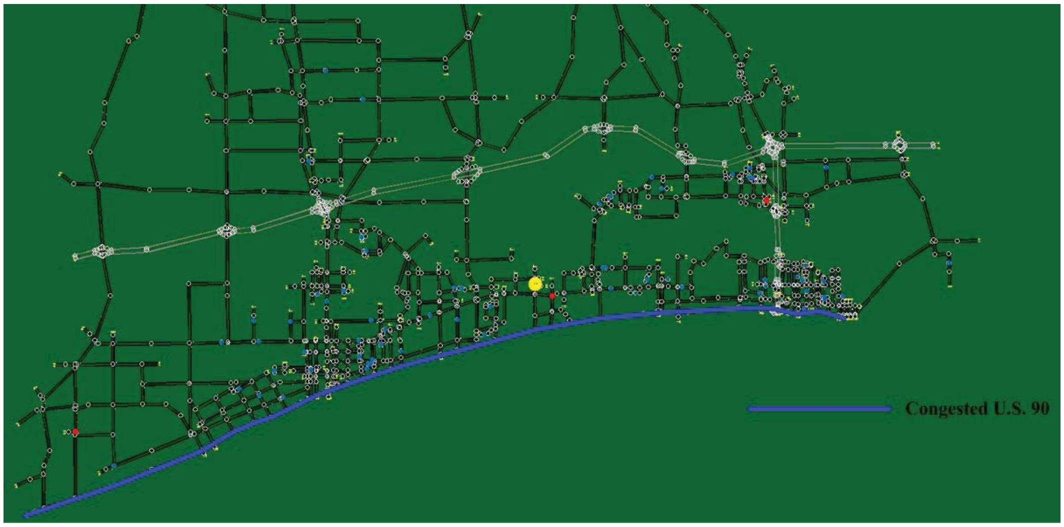


Figure １９ CORSIM Network of Scenario 3(a)

The SmartEvac system is able to capture the travel time surge in real time and update the transit vehicle routes accordingly. Assuming that the travel time surge happens in t_1, the updated results comparing with the results from Scenario 2 are presented as follows.

Results in Scenario 3(a) t_1: the total travel time is 408.6 minutes and the computation time is 173 seconds.

Route 1: cost = 74.2 minutes and load = 21

Dummy Node - Node 27 - Node 18 - Node 8 - Node 14 - Node 15 - Node 19 - Node 22 - Node 7 - Node 66 - Node 25 - Node 1

Route 2: cost = 40.0 minutes and load = 16

Dummy Node - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - Node 62 - Node 1

Route 3: cost = 33.7 minutes and load = 28

Dummy Node - Node 23 - Node 10 - Node 12 - Node 5 - Node 13 - Node 20 - Node 24 - Node 3

Route 4: cost = 44.7 minutes and load = 29

Dummy Node - Node 59 - Node 47 - Node 36 - Node 68 - Node 32 - Node 48 - Node 37 - Node 3

Route 5: cost = 36.8 minutes and load = 30

Dummy Node - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 70 - Node 1

Route 6: cost = 79.1 minutes and load = 29

Dummy Node - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - Node 34 - Node 49 - Node 33 - Node 55 - Node 42 - Node 53 - Node 51 - Node 65 - Node 64 - Node 2

Route 7: cost = 42.1 minutes and load = 30

Dummy Node - Node 41 - Node 31 - Node 35 - Node 40 - Node 52 - Node 30 - Node 46 - Node 58 - Node 45 - Node 43 - Node 54 - Node 56 - Node 38 - Node 2

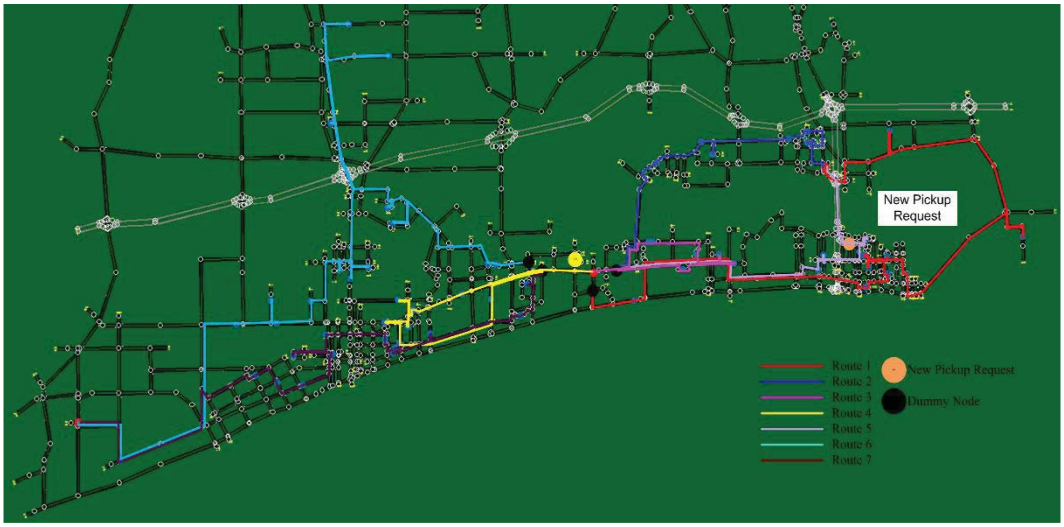


Figure ２０Transit Vehicle Routes after Re-optimization in Scenario 3(a) Interval t1

Three routes—Route 1, Route 4, and Route 5—are revised due to congestions on U.S. 90. See Figure ２１– Figure ２３for a comparison of the results between Scenario 2 and Scenario 3(a).

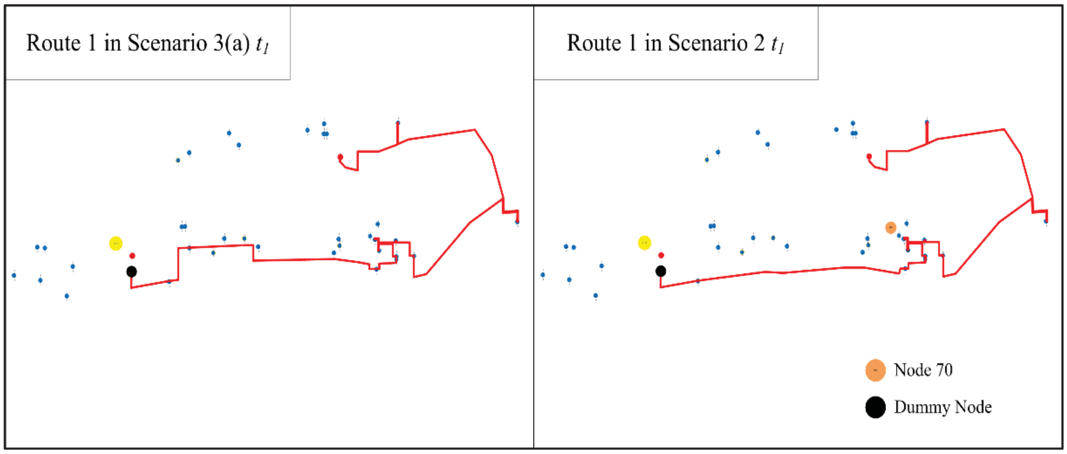


Figure ２１ Comparison of Route 1 between Scenario 2 t1 and Scenario 3(a) t1

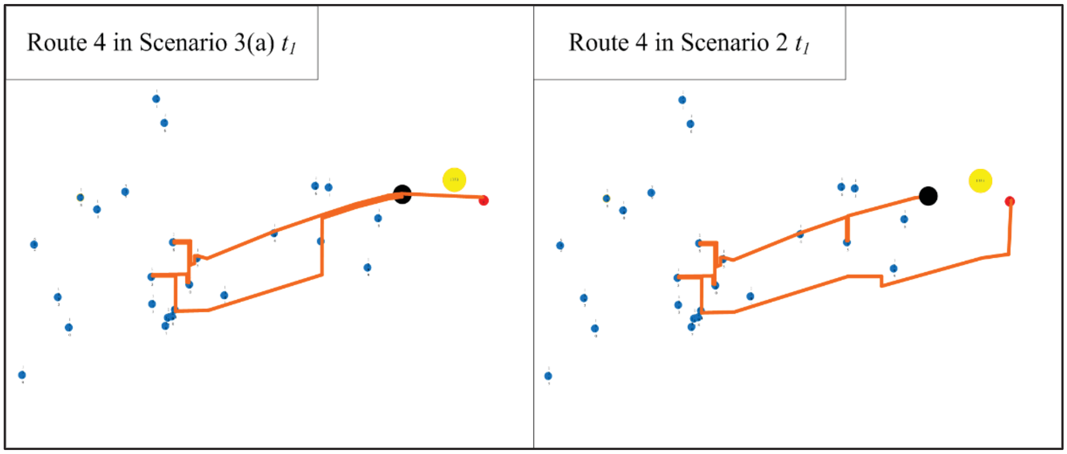


Figure ２２ Comparison of Route 4 between Scenario 2 t1 and Scenario 3(a) t1

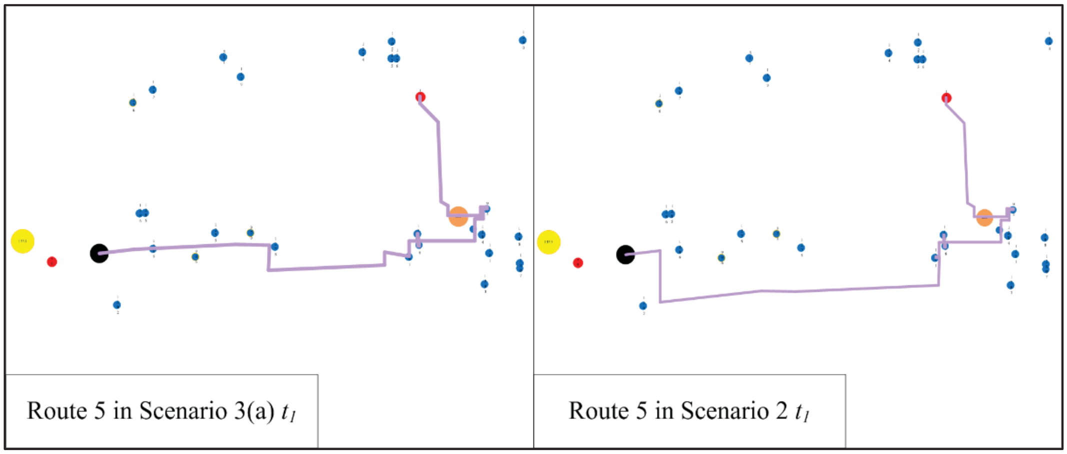


Figure ２３ Comparison of Route 5 between Scenario 2 t1 and Scenario 3(a) t1

In response to the congestions of U.S. 90, the SmartEvac system re-optimize the transit vehicle routes in real time. Since Route 1's travel time will increase from 65.6 minutes to 81.7 minutes due to the congestions of U.S. 90, Route 1's section of U.S. 90 from Eisenhower Drive to Bellman Street is detoured at Beauvoir Road. Vehicle 1 will be diverted to Pass Road, Irish Hill Drive, and Howard Avenue, which are parallel to U.S. 90. The travel time of Route 1 decreases from 81.8 minutes to 74.2 minutes through the detour. Similarly, Route 4's section of U.S. 90 from Tegarden Road to Eisenhower Driver is detoured at Tegarden Road. Vehicle 4 will be diverted to Pass Road. The travel time of Route 4 decreases from 50.3 minutes to 44.7 minutes. Route 5's section of U.S. 90 from Beauvoir Road to Porte Avenue is detoured at Beauvoir Road. Vehicle 5 will be diverted to Pass Road and Irish Hill Drive. The travel time of Route 5 will reduced from 43.7 minutes and 36.8 minutes. In summary, the total travel time saved from the detour on Routes 1, 4, and 5 is 20.1 minutes.

The rest of results in scenario 3(a) from t_3 to t_{30} are listed in *Table 7*.

Table 7 Results of Scenario 3(a)

Time Interval	Total Cost (*Minute*)	No. of Vehicle	Computation Time (*Second*)
t_0	417.9	7	159
t_1	407.3	7	173
t_2	391.6	7	178
t_3	362.7	7	150
t_4	333.7	7	128
t_5	310.2	7	57
t_6	298.9	7	43
t_7	257.1	7	48
t_8	248.2	6	45
t_9	231.7	5	27
t_{10}	207.4	5	13
t_{11}	176.5	5	8

t_{12}	160.4	5	3
t_{13}	157.8	5	1
t_{14}	174.8	5	2
t_{15}	165.1	5	4
t_{16}	158.9	4	2
t_{17}	146.9	4	2
t_{18}	145.8	4	3
t_{19}	134.5	4	3
t_{20}	127.1	4	2
t_{21}	130.5	5	2
t_{22}	122.5	4	1
t_{23}	115.5	4	1
t_{24}	102.5	3	1
t_{25}	95.7	3	1
t_{26}	101.3	2	1
t_{27}	93.2	2	1
t_{28}	94.2	2	1
t_{29}	110.9	3	1
t_{30}	119.6	3	1

Scenario 3(b)

Scenario 3(b) is developed based on Scenario 3(a), but in addition to the incidents on U.S. 90, the Biloxi Bay Bridge is assumed to be broken from t_1, which corresponds to the actual situation in Hurricane Gustav. The Biloxi Bay Bridge carries U.S. 90 over Biloxi Bay between Biloxi and Ocean Springs, as shown in Figure ２４. Route 1 passes the Biloxi Bay Bridge in Scenario 3(a).

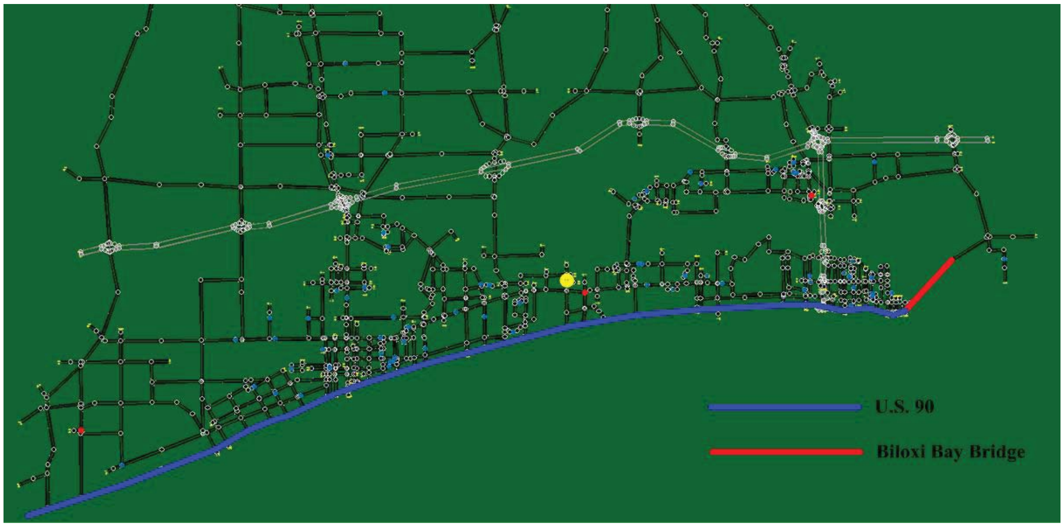


Figure 2 4 CORSIM Network of Scenario 3(b)

The results of Scenario 3(b) in t_1 are summarized as follows. The total travel time is 410.6 minutes and the computation time is 177 seconds.

Route 1: cost = 63.4 minutes and load = 26

Dummy Node - Node 27 - Node 5 - Node 18 - Node 8 - Node 14 - Node 7 - Node 22 - Node 15 - Node 19 - Node 9 - Node 4 - Node 70 - Node 1

Route 2: cost = 71.0 minutes and load = 20

Dummy Node - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - Node 62 - Node 25 - Node 66 - Node 1

Route 3: cost = 15.2 minutes and load = 18

Dummy Node - Node 24 - Node 20 - Node 23 - Node 3

Route 4: cost = 45.1 minutes and load = 29

Dummy Node - Node 59 - Node 47 - Node 36 - Node 68 - Node 32 - Node 48 - Node 37 - Node 3

Route 5: cost = 36.7 minutes and load = 27

Dummy Node - Node 10 - Node 13 - Node 12 - Node 26 - Node 16 - Node 11 - Node 1

Route 6: cost = 110.8 minutes and load = 28

Dummy Node - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - Node 34 - Node 49 - Node 33 - Node 55 - Node 42 - Node 53 - Node 51 - Node 65 - Node 64 - Node 2

Route cost = 68.4 minutes and load = 30

Dummy Node - Node 41 - Node 31 - Node 35 - Node 40 - Node 52 - Node 30 - Node 46 - Node 58 - Node 45 - Node 43 - Node 54 - Node 56 - Node 38 - Node 2

Figure 25 shows the updated transit routes in scenario 3(b) t_3.

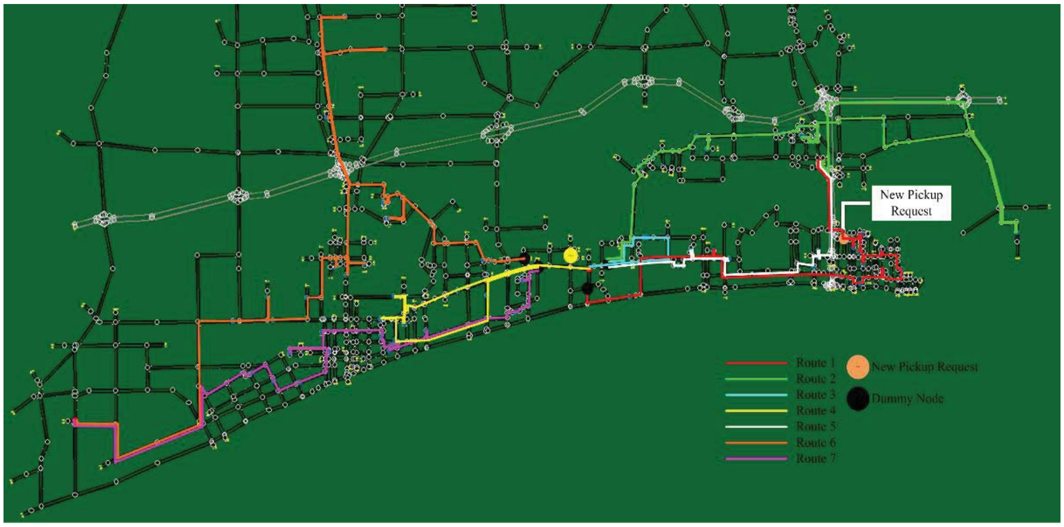


Figure 25 Updated Transit Routes in Scenario 3(b) t3

Four routes—Route 1, Route 2, Route 3, and Route 5—are revised after the destruction of the Biloxi Bay Bridge. See Figure 26 – Figure 29 for a comparison of the results between Scenario 3(a) and Scenario 3(b) in t_1.

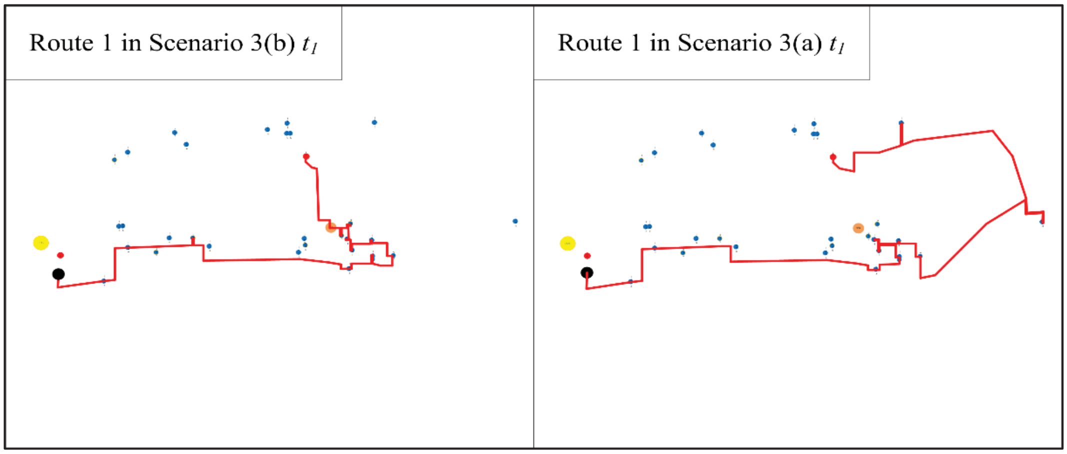

Route 1 in Scenario 3(b) t_1
Route 1 in Scenario 3(a) t_1

Figure ２６ Comparison of Route 1 between Scenario 3(a) t3 and Scenario 3(b) t3

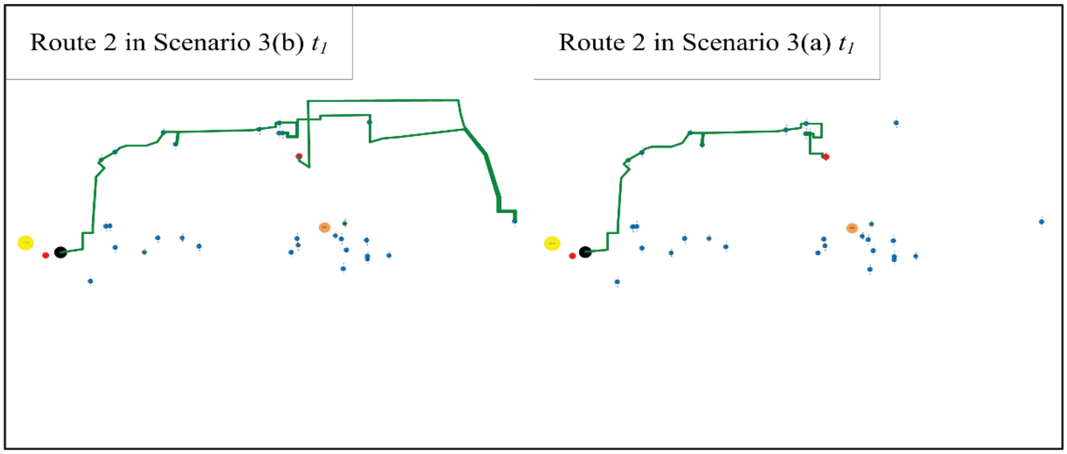

Route 2 in Scenario 3(b) t_1
Route 2 in Scenario 3(a) t_1

Figure ２７ Comparison of Route 2 between Scenario 3(a) t3 and Scenario 3(b) t3

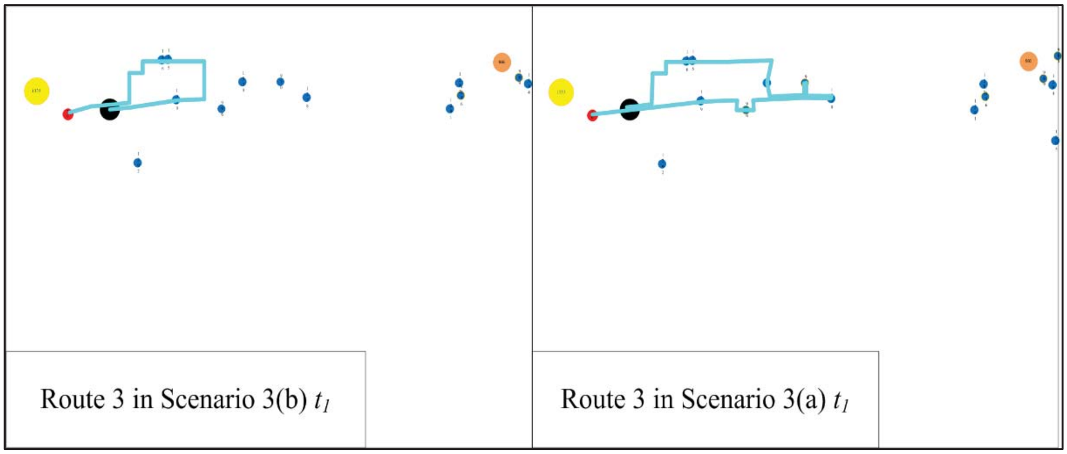

Route 3 in Scenario 3(b) t_1
Route 3 in Scenario 3(a) t_1

Figure ２８ Comparison of Route 3 between Scenario 3(a) t3 and Scenario 3(b) t3

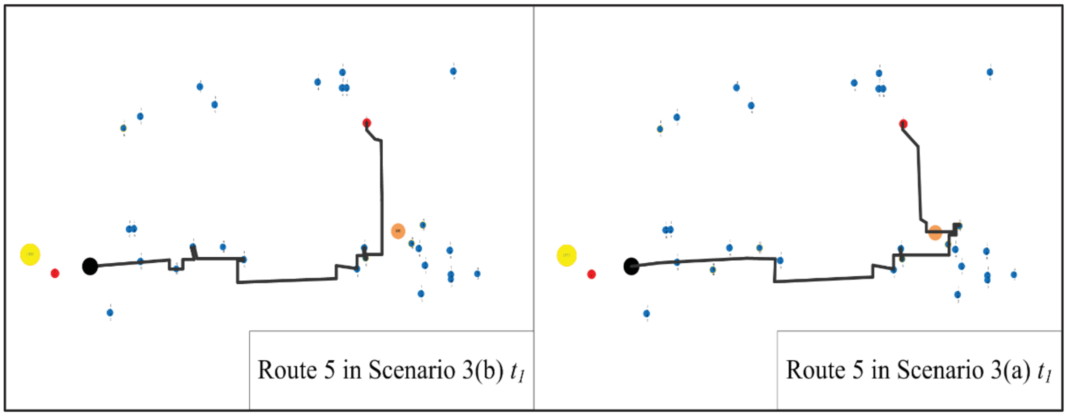

Route 5 in Scenario 3(b) t_1
Route 5 in Scenario 3(a) t_1

Figure ２９ Comparison of Route 5 between Scenario 3(a) t3 and Scenario 3(b) t3

In Scenario 3(b), the Biloxi Bay Bridge is hypothetically broken after the emergency evacuation starts. As a result, Route 1 in Scenario 3(a) is no longer applicable to Node 25 and Node 66. After re-optimization, Node 25 and Node 66 are assigned to Route 2-Vehicle 2, which is the nearest vehicle capable of picking them up. Because Nodes 25 and 66 are removed from Route 1, Vehicle 1 will has sufficient capacity to pick up Node 4, Node 9, and Node 70, which are originally carried by Vehicle 5. The pickup points in Zone 3 are divided into two groups by the Biloxi Bay Bridge and the U.S. 110 Bridge over the Back Bay. The first group, including Nodes 25 and 66, are assigned to Vehicle 2, and the second group, including the rest of nodes in Zone 3, are covered by Vehicle 1. This re-assignment impacts Route 3 and Route 5 as well. The pickup points along with the Pass Road section between Popps Ferry Road and Rodeo Drive are distributed to Routes 3 and 5 optimally.

The rest of results from t_3 to t_{30} are listed in Table 8.

Table 8 Figure 12 Results of Scenario 3(b)

Time Interval	Total Cost (*Minute*)	No. of Vehicle	Computation Time (*Second*)
t_0	417.9	7	161
t_1	410.6	7	177
t_2	394.9	7	180
t_3	365.6	6	156

t_4	337.5	6	135
t_5	314.7	6	54
t_6	301.8	7	50
t_7	260.2	5	47
t_8	251.4	5	44
t_9	237.1	5	31
t_{10}	216.8	5	23
t_{11}	192.5	5	13
t_{12}	172.2	5	9
t_{13}	167.8	4	6
t_{14}	182.5	4	3
t_{15}	169.5	4	3
t_{16}	158.1	3	3
t_{17}	146.4	4	2
t_{18}	147.4	4	2
t_{19}	136.2	4	1
t_{20}	128.9	3	1
t_{21}	132.4	4	1
t_{22}	124.6	4	1
t_{23}	117.1	4	1
t_{24}	104.3	2	1
t_{25}	97.0	2	1
t_{26}	103.1	2	1
t_{27}	94.9	2	1
t_{28}	94.7	2	1
t_{29}	111.5	3	1
t_{30}	120.1	3	1

Results Analysis

Computation Time

The statistics of the computation time for the three scenarios are displayed in Figure ３０. First, the SmartEvac system generates an initial solution using 157 seconds for the network with 74 nodes. Then, the computation time increases as the network size grows with new added pickup points and dummy points. The peak computation times are 171 seconds for Scenario 2, 178 seconds for Scenario 3(a), and 180 seconds for Scenario 3(b), which meets the design standard. The average computation times are 28.9 seconds, 34.3 seconds, and 35.9 seconds, in Scenarios 2, 3(a), and 3(b), respectively. In addition, the computation time shows a similar tendency for all the three scenarios in that a sharp drop occurs from the 5th interval. The primary reason for this phenomenon is that the network size starts to decrease with the completion of part of the pickup requests. For example, in Scenario 2, the network size drops from 76 to 65 in the 5th interval. Another reason is that the initial routes set *R* for each interval is gradually improved with the SmartEvac system running. A high quality initial routes set *R* is able to accelerate the convergence of the column generation algorithm and thus reduce the computation time (Toth et al., 2001).

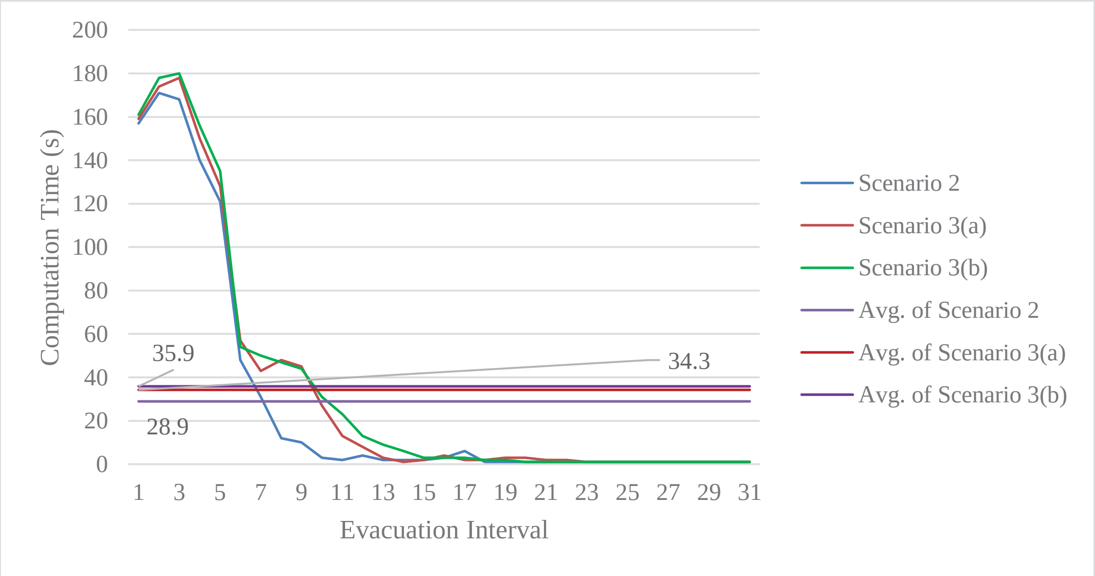


Figure ３０ Computation Time in Scenario 2, Scenario 3(a), and Scenario 3(b)

Response to Evacuation Information Updates

The ability that the SmartEvac system responds to the dynamic evacuation information, such as new pickup requests, is a primary indicator of the SmartEvac system's applicability in a real time emergency evacuation. The system response time is defined as the interval from a new pickup request coming into the system to the implementation of an updated transit vehicle routing plan considering the new pickup request. It is assumed that the arrivals of the new pickup requests are uniformly loaded in the emergency evacuation process. Since the SmartEvac system updates the evacuation information at the end of each interval and re-optimizes the transit vehicle routes in the next interval, the average response time to a new pickup request is $3t/2$ in Scenario 2 with a fixed time interval, where t is the length of the interval. Therefore, the average response time is related with the length of time interval that the SmartEvac system needs to collect dynamic evacuation information and do re-optimization. However, because of the computational burden at the initial stage of the evacuation process, the fixed interval is usually very lengthy; though it becomes redundant when the network size decreases to around 60 nodes. In order to overcome the deficiency with fixed time interval, a dynamic time interval is applied in Scenario 2.

The advantage of a dynamic interval is that its length can be dynamically adjusted based on its previous interval's length and computation time. Figure ３１draws a comparison of response times between Scenario 2 with a fixed interval and Scenario 2 with a dynamic interval. The response time in Scenario 2 with a dynamic interval drops continually until the 8th interval, after which the response time fluctuates around 100 seconds. The average response time with the dynamic interval is 110.8 seconds in contrast with 270.2 seconds with the fixed interval.

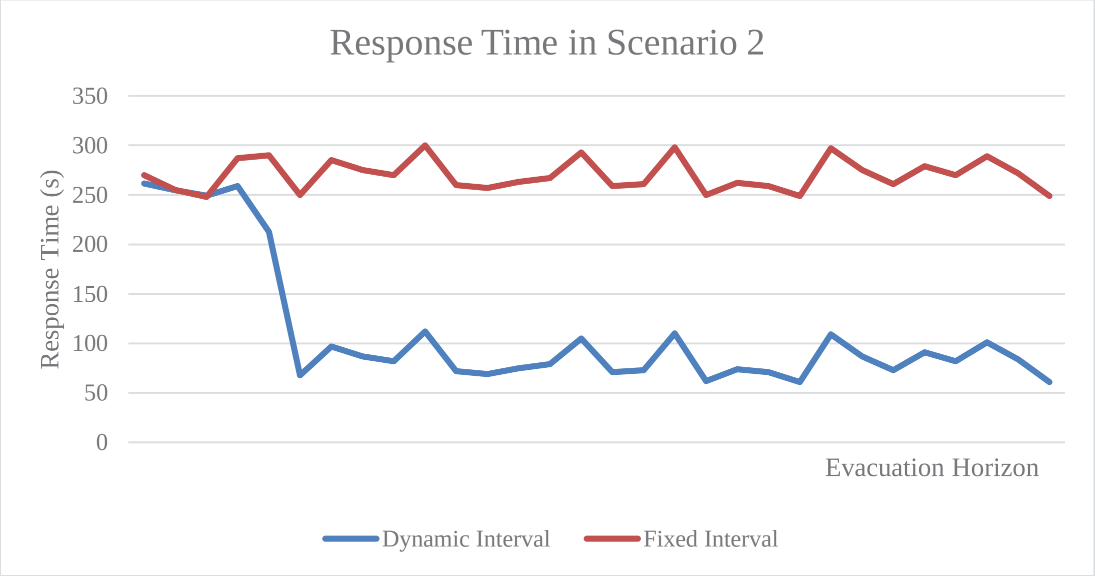


Figure ３ １ Response Time in Scenario 2 with Fixed Interval and Dynamic Interval

Comparison with Real Evacuation Results

An effective way to validate the system is to compare with the results from the real Hurricane Gustav evacuation. During the Hurricane Gustav evacuation, CTA employed 15 transit vehicles and 15 drivers. There were a total of 15 vehicle-trips in the whole process. Table 9 summarized the SmartEvac system’s results and the results from CTA’s record.

Table 9 Comparison of SmartEvac system's results and CTA's record

	SmartEvac	CTA Operations	% Saving
Total Evacuation Time (min)	417.9	637.5	34.4
Average Response Time (min)	1.4	10	86.0
No. of Vehicles Used in the Evacuation	7	15	53.3

The results from the SmartEvac system are much more efficient in terms of the total evacuation time, average response time, and number of vehicles used in the evacuation. The total evacuation time improved 34.4% by the SmartEvac system. Most importantly, the SmartEvac system would respond to a new pickup request under two minutes, while the CTA requires 10 minutes on average. There are only seven vehicles used by the SmartEvac system, which is much less than the 15 used by CTA. All the above results demonstrate that the SmartEvac system significantly outperforms the CTA's old system.

WEB-BASED INTERFACE DEVELOPMENT

The web-based interface design

The Real-Time Transit Vehicle Routing Optimization system should provide an easy and fast-access method to transit agencies and bus drivers. One web interface for SmartEvac is developed and provided. The fundamental requirements of any web interface would include being user-friendly, being able to be accessed anywhere and at any time, and not being computational platform independent (Windows vs. Mac OS). This service receives dynamic evacuation information and traffic information updates and generates the transit operation plan in real time. The website allows end-users access by smart phones, which could send and receive important information for transit drivers. Therefore the web-based application programming interface (API), web-based server host, website, and windows web application are developed and implemented.

The Real-Time Transit Vehicle Routing Optimization web interface would be deployed into two network scenarios. The first is intranet, wherein data between the SmartEvac host server and data from transit agencies are exchanged in the transit center. The second is internet, which contains wireless and wired network connections. The users of the internet network would include drivers who have smart phones and transit agencies who may work outside. Figure ３２ displays the system environment of web interface. It clearly shows the two networks for different roles.

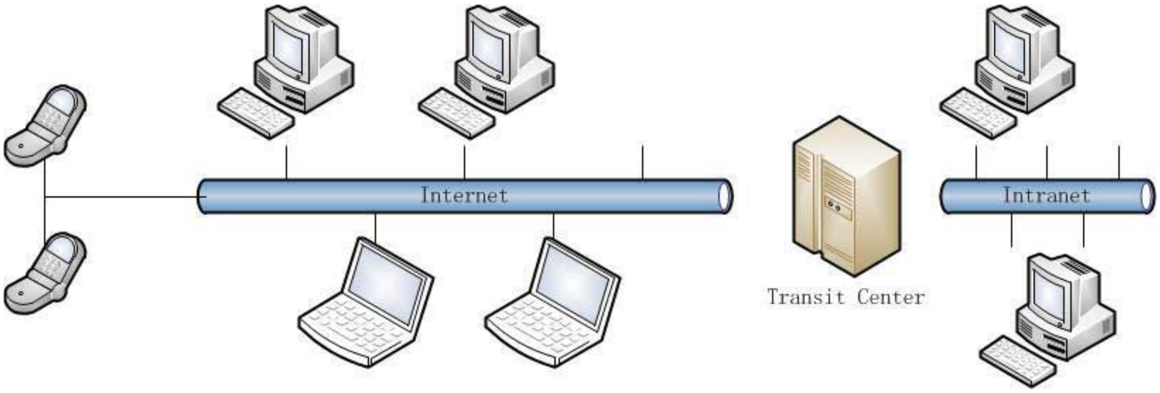


Figure ３２ System environment of web-based interface

Basing one system requirement, the web interface contains three basic parts: interface, server host, and client. Figure ３３ displays a structural diagram of the web interface. In this diagram, web interface basic components are easily identified and understood.

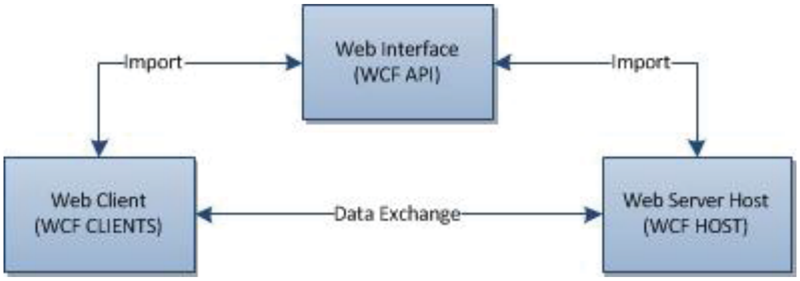


Figure ３３ Structural Diagram

Windows Communication Foundation (WCF) technology perfectly meets the requirement. The WCF is a framework for building service-oriented applications. Using WCF, a software engineer could send data from one application to another at any location with an Internet connection. Furthermore, the WCF service as a new network connection technology has several advantages for this task. WCF service could easily configure between different programming languages, including C++, C#, or VB. In addition, WCF service is not complicated for deployment and configuration. In this web interface, we developed WCF API, WCF Host, and multiple WCF Clients. In WCF HOST and API, future researchers or developers will not need to modify a majority of source code when they need to expand more functions. At WCF Client sides, only few codes could create the connections between client application and host. This is another advantage of WCF.

System Implementation

WCF API

For WCF technology, the basic framework is WCF API, which supports remote call functions for host and clients. Dynamic Link Library (DLL) is chosen for the implementation method for WCF API. It is an interface that contains a number of ports, wherein the main functions of these ports are making connections and transferring data between hosts and clients. The source code below shows the basic definition and implementation methods in WCF API:

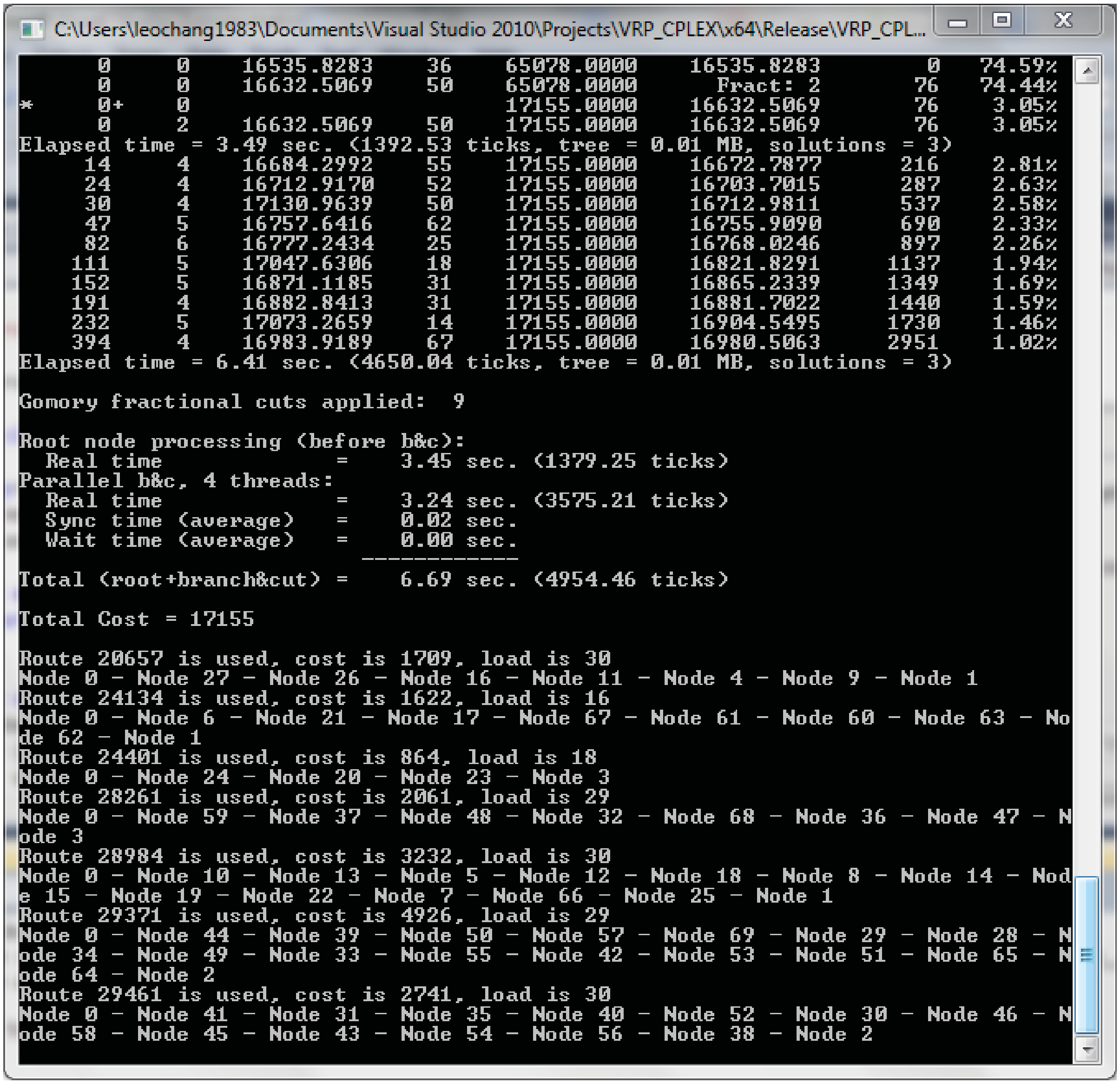
```
    [ServiceContract]
  public interface class ITransferService
  {
        [OperationContract]
        array<TimeInfo>^ GetServerTimeInfo();

        [OperationContract]
        void SetServerTimeInfo(array<TimeInfo>^ serverTimeInfo);
  }
```

The two above methods, GetServerTimeInfo() and SetServerTimeInfo(), are used to exchange data based on WCF API.

WCF Host

The WCF Host is developed as a console application in C++, which could reference IBM CPLEX Optimizer Solver. The WCF Host has two components: the first is loading the IBM CPLEX Solver; the second is building and connecting a website server which could provide the optimization solutions to clients. Figure ３ ４ shows a screenshot of the WCF HOST running time. The application, called VRP_CPLEX, could generate an optimization solution on real-time data.



C:\Users\leochang1983\Documents\Visual Studio 2010\Projects\VRP_CPLEX\x64\Release\VRP_CPL...

```
      0     0    16535.8283    36    65078.0000    16535.8283        0   74.59%
      0     0    16632.5069    50    65078.0000      Fract: 2       76   74.44%
*     0+    0                        17155.0000    16632.5069       76    3.05%
      0     2    16632.5069    50    17155.0000    16632.5069       76    3.05%
Elapsed time = 3.49 sec. (1392.53 ticks, tree = 0.01 MB, solutions = 3)
     14     4    16684.2992    55    17155.0000    16672.7877      216    2.81%
     24     4    16712.9170    52    17155.0000    16703.7015      287    2.63%
     30     4    17130.9639    50    17155.0000    16712.9811      537    2.58%
     47     5    16757.6416    62    17155.0000    16755.9090      690    2.33%
     82     6    16777.2434    25    17155.0000    16768.0246      897    2.26%
    111     5    17047.6306    18    17155.0000    16821.8291     1137    1.94%
    152     5    16871.1185    31    17155.0000    16865.2339     1349    1.69%
    191     4    16882.8413    31    17155.0000    16881.7022     1440    1.59%
    232     5    17073.2659    14    17155.0000    16904.5495     1730    1.46%
    394     4    16983.9189    67    17155.0000    16980.5063     2951    1.02%
Elapsed time = 6.41 sec. (4650.04 ticks, tree = 0.01 MB, solutions = 3)

Gomory fractional cuts applied:  9

Root node processing (before b&c):
  Real time             =    3.45 sec. (1379.25 ticks)
Parallel b&c, 4 threads:
  Real time             =    3.24 sec. (3575.21 ticks)
  Sync time (average)   =    0.02 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    6.69 sec. (4954.46 ticks)

Total Cost = 17155

Route 20657 is used, cost is 1709, load is 30
Node 0 - Node 27 - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 1
Route 24134 is used, cost is 1622, load is 16
Node 0 - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - No
de 62 - Node 1
Route 24401 is used, cost is 864, load is 18
Node 0 - Node 24 - Node 20 - Node 23 - Node 3
Route 28261 is used, cost is 2061, load is 29
Node 0 - Node 59 - Node 37 - Node 48 - Node 32 - Node 68 - Node 36 - Node 47 - N
ode 3
Route 28984 is used, cost is 3232, load is 30
Node 0 - Node 10 - Node 13 - Node 5 - Node 12 - Node 18 - Node 8 - Node 14 - Nod
e 15 - Node 19 - Node 22 - Node 7 - Node 66 - Node 25 - Node 1
Route 29371 is used, cost is 4926, load is 29
Node 0 - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - N
ode 34 - Node 49 - Node 33 - Node 55 - Node 42 - Node 53 - Node 51 - Node 65 - N
ode 64 - Node 2
Route 29461 is used, cost is 2741, load is 30
Node 0 - Node 41 - Node 31 - Node 35 - Node 40 - Node 52 - Node 30 - Node 46 - N
ode 58 - Node 45 - Node 43 - Node 54 - Node 56 - Node 38 - Node 2
```

Figure ３４ Screen Shot of WCF Host Application

The model of this application is the same as the local program explained in previous sections. This application is just the extension of the non-web application. The WCF Host would listen to the client side update and broadcast the latest optimized evacuation plan.

WCF Clients

Based on different targets, end-user terminal applications are developed with two cases: Web-based and Desktop-based. The web-based client can be accessed by computers, laptops, and smart phones that have some kind of an Internet browser. There are no limitations of platforms, operation systems, and software versions. HTTP protocol supports this advantage to end-users. Figure ３ ５ and Figure ３ ６ show two screen shots of the website client. In Figure 4.1, the website is already prepared to connect the server host to receive optimal results.

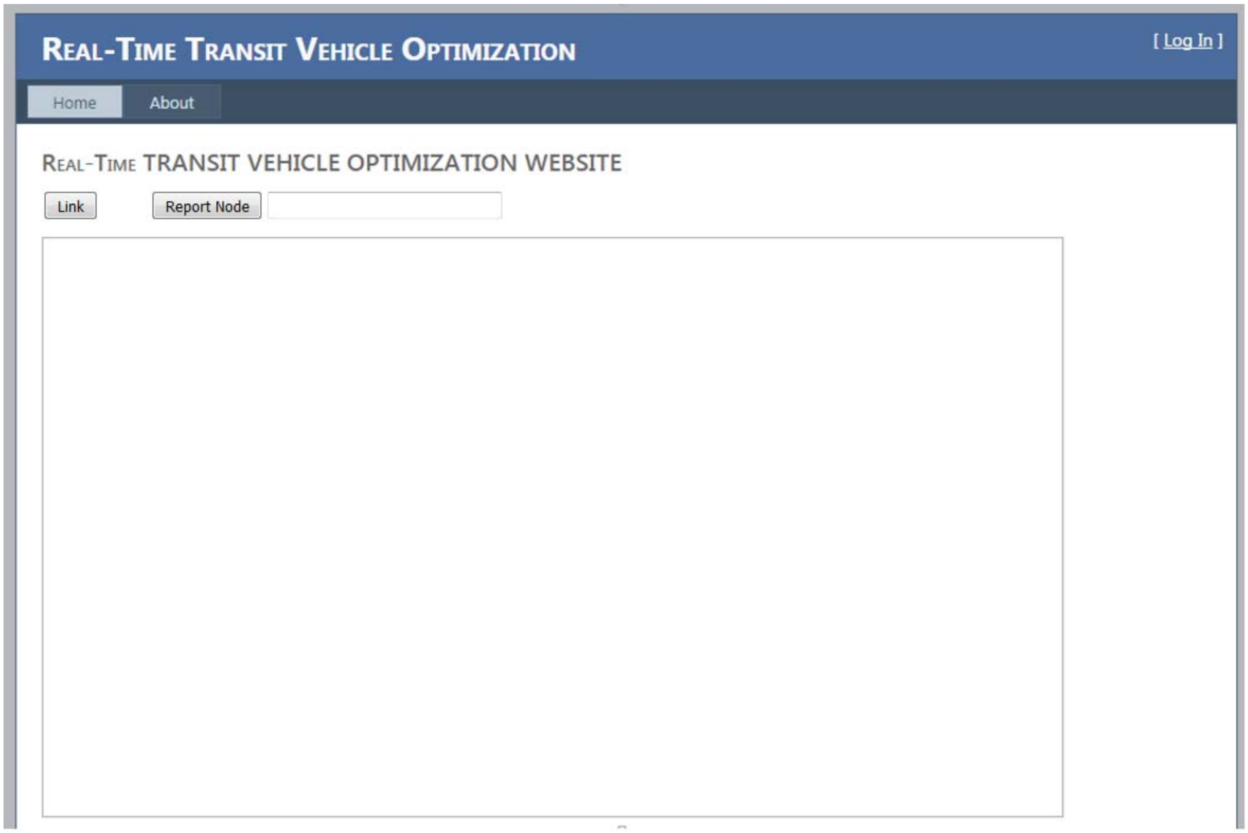
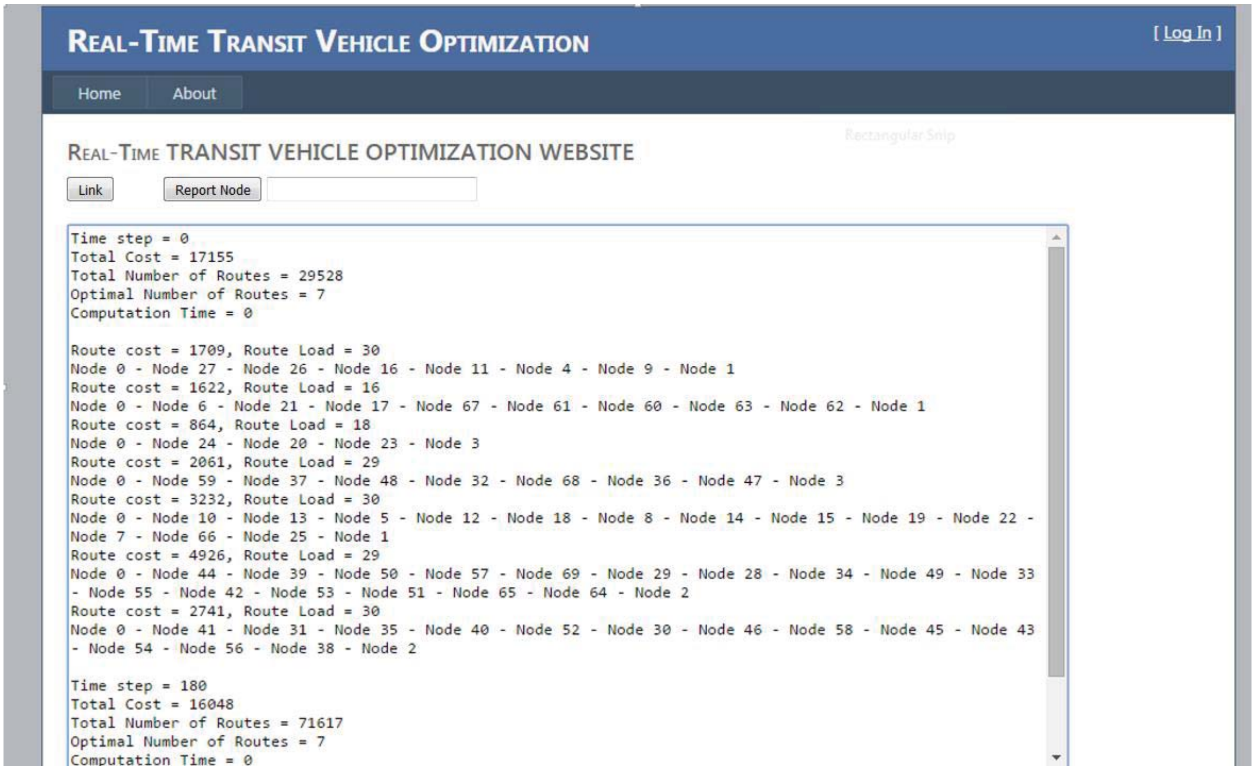

REAL-TIME TRANSIT VEHICLE OPTIMIZATION
[Log In]
Home
About
REAL-TIME TRANSIT VEHICLE OPTIMIZATION WEBSITE
Link
Report Node

Figure ３ ５ Website client is ready for connecting server host

After clicking the Link button on the website, the result of optimization would be provided by the server host and displayed to transit agencies or transit drivers. Figure 4.2 displays the optimal results on the website client.



REAL-TIME TRANSIT VEHICLE OPTIMIZATION
[Log In]
Home
About
REAL-TIME TRANSIT VEHICLE OPTIMIZATION WEBSITE
Link
Report Node

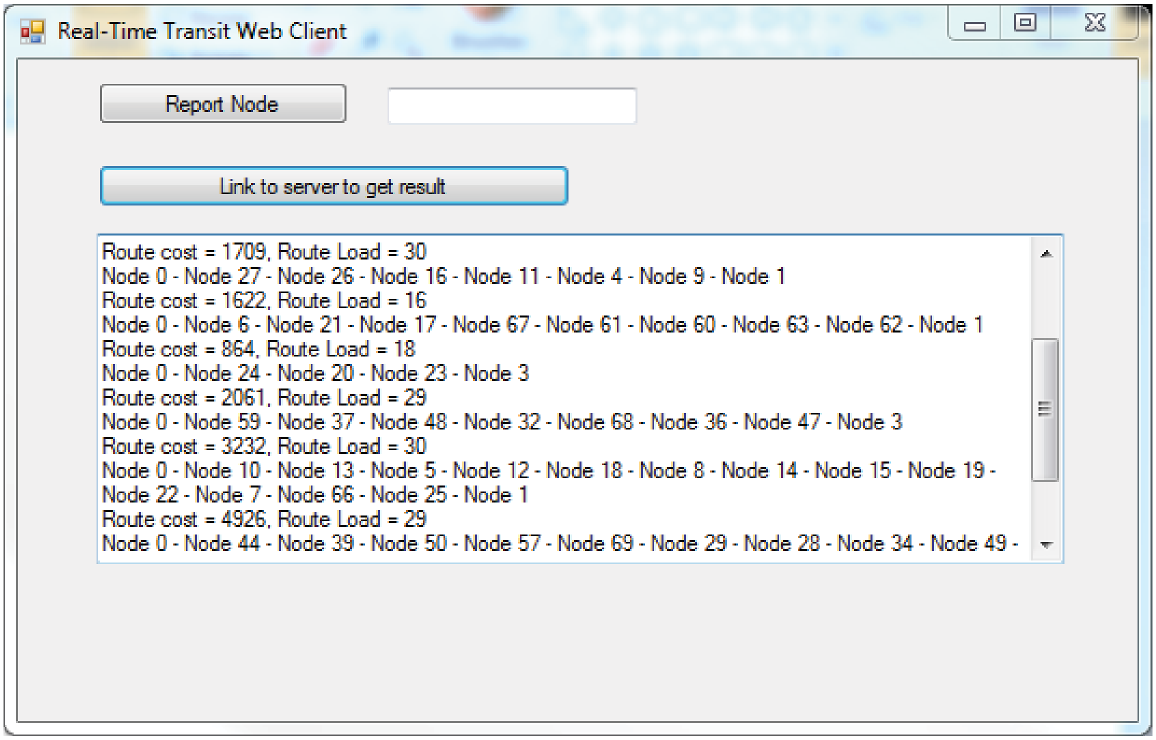
```
Time step = 0
Total Cost = 17155
Total Number of Routes = 29528
Optimal Number of Routes = 7
Computation Time = 0

Route cost = 1709, Route Load = 30
Node 0 - Node 27 - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 1
Route cost = 1622, Route Load = 16
Node 0 - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - Node 62 - Node 1
Route cost = 864, Route Load = 18
Node 0 - Node 24 - Node 20 - Node 23 - Node 3
Route cost = 2061, Route Load = 29
Node 0 - Node 59 - Node 37 - Node 48 - Node 32 - Node 68 - Node 36 - Node 47 - Node 3
Route cost = 3232, Route Load = 30
Node 0 - Node 10 - Node 13 - Node 5 - Node 12 - Node 18 - Node 8 - Node 14 - Node 15 - Node 19 - Node 22 -
Node 7 - Node 66 - Node 25 - Node 1
Route cost = 4926, Route Load = 29
Node 0 - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - Node 34 - Node 49 - Node 33
- Node 55 - Node 42 - Node 53 - Node 51 - Node 65 - Node 64 - Node 2
Route cost = 2741, Route Load = 30
Node 0 - Node 41 - Node 31 - Node 35 - Node 40 - Node 52 - Node 30 - Node 46 - Node 58 - Node 45 - Node 43
- Node 54 - Node 56 - Node 38 - Node 2

Time step = 180
Total Cost = 16048
Total Number of Routes = 71617
Optimal Number of Routes = 7
Computation Time = 0
```

Figure ３ ６ Website client get optimization from server host

Another client type is the desktop-based Windows form client. The significant difference between them is that the desktop-based client is a standalone software application. The result of this feature is that the Windows form client does not need to connect a website server. The Windows desktop client can connect to host servers directly, which would have a higher speed and security than the web-based client.



```
Route cost = 1709, Route Load = 30
Node 0 - Node 27 - Node 26 - Node 16 - Node 11 - Node 4 - Node 9 - Node 1
Route cost = 1622, Route Load = 16
Node 0 - Node 6 - Node 21 - Node 17 - Node 67 - Node 61 - Node 60 - Node 63 - Node 62 - Node 1
Route cost = 864, Route Load = 18
Node 0 - Node 24 - Node 20 - Node 23 - Node 3
Route cost = 2061, Route Load = 29
Node 0 - Node 59 - Node 37 - Node 48 - Node 32 - Node 68 - Node 36 - Node 47 - Node 3
Route cost = 3232, Route Load = 30
Node 0 - Node 10 - Node 13 - Node 5 - Node 12 - Node 18 - Node 8 - Node 14 - Node 15 - Node 19 -
Node 22 - Node 7 - Node 66 - Node 25 - Node 1
Route cost = 4926, Route Load = 29
Node 0 - Node 44 - Node 39 - Node 50 - Node 57 - Node 69 - Node 29 - Node 28 - Node 34 - Node 49 -
```

Figure ３ ７ Windows form client get optimization from server host

In Figure ３ ６ and Figure ３ ７ , end users can input new nodes into the evacuation plan and use the report node button to upload the new node to the optimization host server. The optimization host server would update the new evacuation plan in real time. Both of the clients could transfer routing information based on re-optimized solutions.

CONCLUSION AND RECOMMENDATIONS

This dissertation developed a SmartEvac system for real time transit vehicle routing optimization in an emergency evacuation. The objective of the SmartEvac system is to reduce the total travel time of all transit vehicles. A column generation based CDVRPPD model is integrated into the SmartEvac system. In a static scheme, the difference between the CDVRPPD model and traditional VRP model is that transit vehicles have to deliver the evacuees to a shelter instead of the depot where they depart. Therefore, additional constraints are added to the CDVRPPD model for pickup and delivery. In a real-time scheme, the model is reformulated in each interval over the planning horizon. Essentially, the dynamic model can be converted from a multi-depot CDVRPPD to a single-depot CDVRPPD by introducing dummy pickup points. The conversion can obviously reduce the complexity of the CDVRPPD model. Furthermore, dynamic intervals, whose interval lengths are determined based on the computational performance of the last interval, are implemented over the planning horizon. A case study has demonstrated that the response time of the SmartEvac system can be greatly improved through the implementation of dynamic intervals.

The CDVRPPD model is formulated in a set covering form. The set covering model typically contains an exponential number of variables, making it impractical to solve directly. Therefore, a column generation method—which progressively expands the routes set towards the optimum solution instead of enumerating all the routes—is applied to solve the model. The column generation operation is based on a master-problem and sub-problem structure. The master problem model guides the routes set expansion, while the sub-problem model is developed to price out all of the routes necessary to construct an optimal solution. In a real-time scheme, the initial routes set is generated by integrating a Clarke-Wright saving algorithm with insertion heuristic. The routes set from the last interval is revised to be part of the initial routes set of the current interval. The computational results indicate that the average improvement of lower bound reaches 12.5% on the benchmark problems in comparison with Agarwal, Mathur, and Salkin (1989), Bixby (1998), and Hadjiconstantinou, Christofides, and Mingozzi (1995)'s results in the literature. In addition, the computation time still locates in an affordable range for a real-time system when dealing with the clustered benchmark problems with a network size of 50–100. The increase of computational time by introducing cycle elimination reveals that the system is suitable for a network of a size around 100.

A case study based on the Hurricane Gustav evacuation process is used to demonstrate the SmartEvac system in real scenarios. CORSIM simulations are developed to provide data for the SmartEvac system. Transportation network data in the Gulf Coast area are collected in a field survey. CORSIM RTE is developed as an interface to exchange data between CORSIM simulation and the SmartEvac system. Different scenarios corresponding to the different situations that happened in the Hurricane Gustav emergency evacuation are proposed to evaluate the performance of the SmartEvac system in response to real-time data. The average processing time is 28.9 seconds, and the maximum processing time is 171 seconds (Scenario 2), which demonstrate the SmartEvac system's capability of real-time vehicle routing optimization.

A Windows Communication Foundation based Client/Server SmartEvac is also completed within this project. This enables remote/web deployment of the systems to any transit agencies worldwide. An API is provided to upload their fleet, vehicle, pickup locations, any changes in pickup location, and travel time. The SmartEvac provides optimized routing information, directly to drivers and transit agencies.

In summary, the major contribution of this dissertation is the development of a SmartEvac system which is able to handle real-time transit vehicle routing in an emergency evacuation of 200–250 evacuees. A traditional VRP model is revised to be applicable in a real-time scheme. The implementation of dynamic intervals could effectively reduce the system response time to an emergency. In addition, the proposed 2-Cycle elimination algorithm could tight the lower bound without contaminating the overall performance of the system.

REFERENCES

Afshar, A. and A. Haghani (2008). "Heuristic Framework for Optimizing Hurricane Evacuation Operations." Transportation Research Record: Journal of the Transportation Research Board 2089(-1): 9-17.

Agarwal, Y., Mathur, K., and Salkin, H.M. A set-partitioning-based exact algorithm for the vehicle routing problem. Networks, 19:731-749, 1989.

ARCADIS U.S., Inc. "Evacuation Time Estimates for Browns Ferry Nuclear Power Plant Plume Exposure Pathway Emergency Planning Zone." TM120006.0001, November 2011.

ARCADIS U.S., Inc. "Evacuation Time Estimates for the Three Mile Island Plume Exposure Pathway Emergency Planning Zone" B0033739.0000, December 2012.

Balas, Egon. "The prize collecting traveling salesman problem." Networks 19.6 (1989): 621-636.

Beasley, J. E., and Nicos Christofides. "An algorithm for the resource constrained shortest path problem." Networks 19.4 (1989): 379-394.

Bhaduri, B., C. Liu, and O. Franzese. "Oak Ridge evacuation modeling system (OREMS): A PC-based computer tool for emergency evacuation planning." Symposium on GIS for Transportation. 2006.

Bixby, A. Polyhedral analysis and effective algorithms for the capacitated vehicle routing problem. Ph.D. dissertation, Northwestern University, Evanston, IL, 1998.

Cassady, C.R., LeMay, S. A., Schneider, K., Starks, D., and Chai, P. (2004). *Cost of Ownership Modeling for Support Equipment at Intermodal Transportation Terminals* . United States Department of Transportation, Research and Special Programs Administration.

Chan, C. P. (2010). *Large Scale Evacuation of Carless People During Short- and Long-Notice Emergency*, PhD Dissertation, Department of Systems and Industrial Engineering, The University of Arizona

Cherry, C., Hickman, M., and Garg, A., 2006. ***Design of a Map-Based Transit Itinerary Planner***. *Journal of Public Transportation* 9(2), 45-68.

Chiu, Yi-Chang, Pavan Korada, and Pitu B. Mirchandani. "Dynamic traffic management for evacuation." Proc. of the 84th Annual Meeting of the Transportation Research Board (CD-ROM), Washington, DC. 2005.

Chiu, Y.-C., H. Z. Hong, et al. (2008). "Evaluating Regional Contra-Flow and Phased Evacuation Strategies for Texas Using a Large-Scale Dynamic Traffic Simulation and Assignment Approach." Journal of Homeland Security and Emergency Management 5(1).

Chiu, Y. C., et al. "DynusT User's Manual." (2010). Available: http://dynust.net/.

Chowdhury, M. (2000). *Intermodal Transit System Coordination with Dynamic Vehicle Dispatching.*. Ph.D. Dissertation, Institute for Transportation, New Jersey Institute of Technology

Chowdhury, M. and Chien S. (2000). Intermodal Transit System Coordination. The Preprint, 79th Annual Meeting, Transportation Research Board, Washington D.C.

Chowdhury, M. and Chien S. (2001). *Optimization of Transfer Coordination for Intermodal Transit Networks.*. 80th Annual Meeting, Transportation Research Board, TRB Paper No. 01-0205, Washington D.C.

Chowdhury, M. and Chien S. (2011). *Joint Optimization of Bus Size, Headyway, and Slack Time for Efficient Timed*, Transfer Journal of Transportation Research Board, TRR No. 2218, pp 48~58, October 2011.

Christofides, N. and Eilon, S. An algorithm for the vehicle dispatching problem. Operational Research Quarterly, 20:309-318, 1969.

Christofides, N., Mingozzi, A., and Toth, P. The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, Combinatorial Optimization, Wiley, Chichester, UK, 1979, pp. 315-338.

Citilabs. "Cube Voyager Operating Manual." Tallahassee FL USA (2013).

Clarke, G., and Wright, J.V. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research, 12:568-581, 1964.

Dantzig, G., and Ramser, J. (1959). The Truck Dispatching Problem. Management Science, 6(1): 80-91.

Dantzig, George B., and Philip Wolfe. "Decomposition principle for linear programs." Operations research 8.1 (1960): 101-111.

Dell'Amico, Mauro, Francesco Maffioli, and Peter Värbrand. "On Prize-collecting Tours and the Asymmetric Travelling Salesman Problem." International Transactions in Operational Research 2.3 (1995): 297-308.

Desrochers, M. “An Algorithm for the Shortest Path Problem with Resource Constraints.” Technical Report G-88-27, GERAD, 1988.

Desrochers, M., Desrosiers, J., and Solomon, M. “A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows.” Operations Research, Volume 40, Number 2, Page 342-354, April 1992.

Dror, M. (1994). Note on the complexity of the shortest path models for column generation in VRPTW. Operations Research 42:977-978.

Feillet, D., Dejax, P., Gendreau, M., Gueguen. C. An Exact Algorithm for the Elementary Shortest Path Problem with Resource Constraints: Application to Some Vehicle Routing Problems. Networks, Volume 44, Issue 3, Pages 216–229, October 2004.

Feillet, Dominique, Pierre Dejax, and Michel Gendreau. "Traveling salesman problems with profits." Transportation science 39.2 (2005): 188-205.

FEMA. (August 28, 2013). HURREVAC 2013. Retrieved 2014, from http://www.hurrevac.com/.

B. Ferris, K. Watkins, and A. Borning, *Onebusaway : A transit traveller information system*, in Proc. of Mobicase 2009. SanDiego, USA, 2009, pp. 92 – 106.[SS 06]

Goel, A. and V. Gruhn (2005). "Solving a dynamic real-life vehicle routing problem." Operations Research Proceedings 2005: 367-372.

Golden, Bruce L., Larry Levy, and Rakesh Vohra. "The orienteering problem." Naval research logistics 34.3 (1987): 307-318.

Graham, D.W., C.R. Cassady, R.O. Bowden, and S.A. LeMay (2000), *Modeling Intermodal Transportation Systems: Establishing a Common Language*, Transportation Law Journal, pp. 55-68.

Hadjiconstantinou, E., Christofides, N., and Mingozzi, A. A new exact algorithm for the vehicle routing problem based on *q*-paths and *k*-shortest paths relaxations. Annals of Operations Research, 61:21—43, 1995.

Han, Lee D., and Fang Yuan. "Evacuation modeling and operations using dynamic traffic assignment and most desirable destination approaches." 84th Annual Meeting of the Transportation Research Board, Washington, DC. 2005.

Hobeika, A.G., and B. Jamei, 1985. “MASSVAC: A Model for Calculating Evacuation Times under Natural Disaster.” Emergency Planning, Simulation Series, Vol. 15, pp. 23-28.

Hobeika, A.G., and Chang Kyun Kim, “Comparison of Traffic Assignments in Evacuation Modeling”, IEEE Transactions on Engineering Management, Vol. 45, No. 2, May 1998.

R. Hoar, *Visualizing Transit Through a Web Based Geographic Information System,* in International Journal of Environmental, Ecological, Geological and Marine Engineering Vol:2 No:10, 2008, World Academy of Science, Engineering and Technology

R. Hoar, *A personalized web based public transit information system with user feedback*, in 13th International IEEE Conference on Intelligent Transportation Systems (ITSC) ,Madeira Island, Portugal, 2010, pp. 1807 – 1812

Houck, D.J., J.C. Picard, M. Queyranne, R.R. Vemuganti. 1980. The travelling salesman problem as a constrained shortest path problem: Theory and Computational Experience. Operation Research 17, 93-109.

Hunter, J. Stuart. "The exponentially weighted moving average." J. QUALITY TECHNOL. 18.4 (1986): 203-210.

Irnich, Stefan, and Daniel Villeneuve. "The shortest-path problem with resource constraints and k-cycle elimination for k≥ 3." INFORMS Journal on Computing 18.3 (2006): 391-406.

Jin, Mingzhou, Kai Liu, and Burak Eksioglu. "A column generation approach for the split delivery vehicle routing problem." Operations Research Letters 36.2 (2008): 265-270.

KLD Associates INC., 1984. “Formulations of the DYNEV and IDYNEV Traffic Simulation Models Used in EESF.” Rep. Prepared for the Federal Emergency Management Agency, Commack, NY.

Kwon, Eil. Development of Operational Strategies for Travel Time Estimation and Emergency Evacuation on a Freeway Network. No. MN/RC-2004-49. 2004.

Larsen, Jesper. Parallelization of the vehicle routing problem with time windows. Diss. Technical University of Denmark, Department of Informatics and Mathematical Modeling, 1999.

Larsen, Allan, and Oli BG Madsen. The Dynamic Vehicle Routing Problem. Dissertation. Technical University of Denmark, Department of Transportation, Logistics & ITS, 2000.

Lim, Yu Yik. “Modeling and Evaluating Evacuation Contraflow Termination Point Designs.” Master's Thesis. Louisiana State University, 2003.

Liu, S. and S. Lee (2003). "A two-phase heuristic method for the multi-depot location routing problem taking inventory control decisions into consideration." The International Journal of Advanced Manufacturing Technology 22(11): 941-950.

Mahmassani, Hani S., Hu, Ta-Yin, Peeta S., and A. Ziliaskopoulos. Development and testing of dynamic traffic assignment and simulation procedures for atis/atms applications. Report DTFH61-90-R-00074-FG, U.S. DOT, Federal Highway Administration, McLean, Virgina, 1994.

Mahmassani, Hani S., Hayssam Sbayti, and Xuesong Zhou, DYNASMART-P, Intelligent Transportation Network Planning Tool, Version 1.0 User's Guide, Maryland Transportation Initiative, University of Maryland, College Park, September 2004.

McTrans, (2014). McTrans Center » University of Florida. [Online] Mctrans.ce.ufl.edu. Available at: http://mctrans.ce.ufl.edu/mct/ [Accessed 28 Apr. 2014].

Naser, Mohammad, and Shawn C. Birst. Mesoscopic Evacuation Modeling for Small-to Medium-Sized Metropolitan Areas. No. MPC Report No. 10-222. Mountain-Plains Consortium, 2010.

Peng, Z.R., and Huang, R., 2000. *Design and Development of Interactive Trip Planning for Web-Based Transit Information Systems*. *Transportation Research Part C* 8(5), 2 409-425.

Perkins, J., I. Dabipi, et al. (2001). Modeling Transit Issues Unique to Hurricane Evacuations: North Carolina's Small Urban and Rural Areas, Greensboro, NC Urban Transit Institute/Transportation Institute, NC A&T State University.

Righini, G., Salani, M. New Dynamic Programming Algorithms for the Resource-Constrained Elementary Shortest Path Problem. Networks 51.3 (2008): 155-170.

Schwardt, M. and J. Dethloff (2005). "Solving a continuous location-routing problem by use of a self-organizing map." International Journal of Physical Distribution and Logistics Management 35(6): 390.

Schomborg, A., K. Nökel, and A. Seyfried. "Evacuation assistance for a sports arena using a macroscopic network model." Pedestrian and Evacuation Dynamics. Springer US, 2011. Pp. 389-398.

Secomandi, N. (2000). "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands." Computers and Operations Research 27(11-12): 1201-1225.

Sheffi, Y., Mahmassani, H., and Powell, W.B., 1982, “A Transportation Network Evacuation Model.” Transportation Research, Part A, 16(3), pp. 209–218.

Songchitruksa, Praprut, et al. "Dynamic traffic assignment evaluation of hurricane evacuation strategies for the Houston-Galveston, Texas, region." Transportation Research Record: Journal of the Transportation Research Board 2312.1 (2012): 108-119.

Sun, D., Peng, Z.R., Kong, C. and Chen, W., *Development Of Web-Based Transit Trip Planning System Based on the Service Oriented Architecture*, Transportation Research Record Journal of the Transportation Research Board, 12/2011, DOI: 10.3141/2217-11.

Tagliaferri, Anthony Paul. "Use and comparison of traffic simulation models in the analysis of emergency evacuation conditions." Master's Thesis. North Carolina State University, 2005.

Theodoulou, Gregoris, and Brian Wolshon. "Alternative methods to increase the effectiveness of freeway contraflow evacuation." Transportation Research Record: Journal of the Transportation Research Board 1865.1 (2004): 48-56.

Toth, P., Paolo, and Daniele Vigo. The vehicle routing problem. Society for Industrial and Applied Mathematics, 2001.

Wang, Feng, et al. "Study of the Effects of Evacuation Routes and Traffic Management Strategies in Short-notice Emergency Evacuation in Downtown Jackson." Transportation Research Board 93rd Annual Meeting. No. 14-4920. 2014.

Wen, Y., Zhang, L., and Huang, Z. 2012. “Coordination of Connected Vehicle and Transit Signal Priority in Transit Evacuations.” Transportation Research Board Annual Meeting 2012 Paper #12-3152.

Yuan, F., L. D. Han, S. Chin., and H. Hwang, 2006, “Proposed Framework for Simultaneous Optimization of Evacuation Traffic Destination and Route Assignment” Journal of the Transportation Research Board, No. 1964, Transportation Research Board of the National Academies, Washington, DC, pp 50–58.

Yue Liu and Jie Yu (2012). *Emergency Evacuation Planning for Highly Populated Urban Zones: A Transit-Based Solution and Optimal Operational Strategies*, Emergency Management, Dr. Burak Eksioglu (Ed.), ISBN: 978-953-307-989-9

Zhang, Li, et al. "Simulation Modeling and Application with Emergency Vehicle Presence in CORSIM." Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th. IEEE, 2009.

Zhang, Li, Yi Wen, and Minzhou Jin. The Framework for Calculating the Measure of Resilience for Intermodal Transportation Systems. No. NCIT Report# 10-05-09. 2009.

Zheng, Hong, et al. "Modeling of evacuation and background traffic for optimal zone-based vehicle evacuation strategy." Transportation Research Record: Journal of the Transportation Research Board 2196.1 (2010): 65-74.

Zou, Nan, et al. "Simulation-based emergency evacuation system for Ocean City, Maryland, during hurricanes." Transportation Research Record: Journal of the Transportation Research Board 1922.1 (2005): 138-148.