A SIMULATION FRAMEWORK 
FOR MODELING LARGE-SCALE FLEXIBLE TRANSIT SYSTEMS

Jaeyoung Jung
Assistant Project Scientist, Ph.D.
Institute of Transportation Studies
University of California, Irvine
Irvine, CA 92697-3600
Email: jaeyounj@uci.edu

R. Jayakrishnan
Professor, Ph.D.
Institute of Transportation Studies
Department of Civil and Environmental Engineering
University of California, Irvine
Irvine, CA 92697-3600
Email: rjayakri@uci.edu
A SIMULATION FRAMEWORK FOR MODELING LARGE-SCALE FLEXIBLE TRANSIT SYSTEMS

ABSTRACT

Despite the prevalent use of simulation methods in the majority of areas in transportation research, transit and paratransit research has not heavily relied on simulation models, not only because traditional transit studies do not necessarily require simulation techniques, but also because there are no proper simulation packages available for modeling innovative transit systems. This study proposes a new type of simulation framework targeting large-scale flexible transit systems with various vehicle operation schemes. The intent of this study is to describe the generalized concepts and the detailed architecture of the framework developed for urban transportation networks. A comprehensive process for the framework development is discussed, including the considerations in simulation data conversion and user interface design. Two different real-time flexible transit applications are modeled using the proposed simulation framework: High Coverage Point-to-Point Transit (HCPPT) and real-time Shuttle service. The detailed simulation results are provided by addressing the importance of simulation design and operational features. The results also show that the simulation framework can model realistic large networks several times faster than real-time with standard desk-top computers, raising the possibility of its use in real-time optimization schemes, as well as for synchronous modeling with commercial road traffic simulators.
1. INTRODUCTION

The use of simulation has become prevalent in transportation systems modeling due to its ability to analyze and assess system performance and complicated infrastructural designs. Despite the promise of simulation methods, transit and paratransit research has not heavily relied on their use. Since conventional transit vehicles move on fixed-routes and use fixed-stops, traditional transit studies do not necessarily require simulation techniques at the network level. If simulation is needed for such traditional transit, it is also relatively easy to accomplish it with many of the existing simulation software, even if they were primarily designed to model network level auto traffic. Moreover, many demand responsive transit (DRT) and paratransit studies have focused on conventional ‘dial-a-ride’ transit systems in which passengers make reservations a day prior to their travel for vehicles with flexible routes. Generally, rather than simulating the operation of DRT systems, modeling ‘dial-a-ride’ systems makes use of optimization formulations which consider vehicle routes and passenger time windows. It is the real-time transit systems that require detailed simulation studies, as the efficiency of such systems depends on the effects of congestion and other dynamic changes in the network system during study periods within a day. Many such real-time details are difficult to incorporate in the analytical models made for such systems. This paper focuses on the development of simulation frameworks for this purpose.

Though significantly less than the use of simulation in traffic network studies, there have been research that used simulations for traditional transit system studies in the past decades, and specifically in the use of commercial software such as PARAMICS (1), Vissim (2), Aimsun (3), DYNASMART-P (4), and TransModeler (5). Some examples can be found in (6, 7, 8, 9, 10). In recent years, simulation has also become an attractive approach for investigating and modeling advanced fixed route transit systems such as Light Rail Transit (LRT) and Bus Rapid Transit (BRT) due to model complexities and non-linear system performance, for which microscopic traffic simulation is known as an ideal alternative. A simulation of Light Rail Transit (LRT) using the TRAF-Network Simulator (NETSIM) program and JRH Transportation Engineering’s TransSim II tools was conducted by Venglar et al. in (11). In that study, they showed the simulation model could produce moderately accurate estimation within the studied network compared with real-world data. A computer simulation model for Bus Rapid Transit (BRT), SmartBRT, was developed by Balvanyos et al. based on PARAMICS in (6). The purpose of this study was not only to evaluate the operational concepts for BRT, but also to visualize the impact of the infrastructure.

For flexible transit systems such as Demand Responsive Transit (DRT), real-time shared-ride systems, and other flexible passenger door-to-door services, the use of simulation modeling is necessary because solutions for these problems can be characterized as a combination of dynamic vehicle routing problems and subsequent individual vehicle operations control. Evaluation of such solutions cannot be performed by analytical methods as the models grow more elaborate with higher complexity. Cortés et al. in (7) developed an integrated microscopic simulation framework for High Coverage Point-to-Point Transit (HCPPT) by employing PARAMICS and its Application Programming Interface (API) Plug-ins. This study demonstrated the level of vehicle operation flexibility that can be brought into a microscopic simulation package for larger urban networks. The framework therein uses a unique approach for communication, integration, and routing in which all path-based decisions consider a simplified network topology, e.g. an abstract network (ABSNET) that is used to keep the behavior of all transit vehicles at the aggregated level. The main reason was that the network path processing algorithms show a nonlinear increase in computational requirements as network size increase in most microscopic simulation levels. An additional communication module was required to translate every network attribute from one level of aggregation to the other, which can cause discrepancy between dispatch decision and simulation levels.

In order to apply existing microscopic simulation frameworks to transit, detailed information concerning transportation network, transit routes, and stops are also required. For instance, a stop may be at some
point along a network link, and the path-finding algorithms of the simulation software may be primarily based on links and nodes. If the model can handle the transit system details to some extent, it is still possible that much input data will need to be gathered by manually matching drawings as the simulation framework itself does not include concepts such as map-based input data.

The most important drawback of existing transportation network simulation software is that their focus is on simulating the personal automobiles, rather than on modeling any flexible transit operation. Thus most microscopic simulation packages typically require parameter calibration and dynamic origin-destination (OD) demand estimation of zonal trips, whereas flexible transit simulation needs to handle passenger demand generation and assignment to the transit system at a trip-level with additional parameters for each passenger. That is, the existing software typically uses the zone-to-zone trip tables and generates demands randomly for zones, often assigning them randomly to entry points of auto traffic within the zones.

Usually, vehicles that are released from origin zones have fixed destinations and necessarily arrive at their destination zones, not constantly moving around the network, which could be a prohibitive characteristic for simulating dynamic vehicle operations. In other words, the flexible transit demand generation and associate vehicle operation require fundamentally different processes, and the zone and centroid structures are rarely sufficient for the modeling.

To overcome the limitations of existing simulation packages, some past studies have attempted the development of customized simulation methods. Fu developed a simulation framework based on information technologies such as Automatic Vehicle Location (AVL) and digital telecommunication in (12). Although the simulation exercise was carried out for a hypothetical geographic area and an artificial road network, the simulation was able to show that an online scheduling system is efficient enough to dispatch vehicles responding to real-time operational events such as new trip requests and vehicle breakdown. Quadrifoglio et al. in (13) used a simulation model to investigate the effect of a zoning strategy on performance measures based on data for DRT service in Los Angeles County. For their study, real paratransit demand data provided by a local agency was used to generate random sample demands, but a simple representative network was used in place of the actual network due to lack of available information. Deflorio in (14) described a strips simulation method for demand responsive systems in order to assess the efficiency of the proposed system. Various trip requests scenarios are considered for the assess trip-plans by recreating an actual demand structure on aggregated zonal data. Cheng and Nguyen in (15) developed a multi-agent simulation platform for evaluating taxi fleet operations, TaxiSim, to evaluate the impact of fleet management policies. They proposed the simulation architecture by designing taxi agent’s strategy. However, TaxiSim involves the macro-level regularity rather than utilizing the micro-level since their interest was the revenue accumulation pattern over time such as day-to-day patterns.

In contrast to many transit simulation studies, real-time flexible transit systems employ medium to large sized vehicles to provide door-to-door services as well as dynamic vehicle schedules on flexible routes. Hence, such simulation should be able to accommodate dynamic demand variations and consider detailed spatial and temporal characteristics of individuals. Moreover, the vehicle operation schemes used in the simulation should include great flexibilities and details with regards to vehicle routing algorithms and vehicle controls. For these reasons, this study proposes a new type of simulation concept for flexible transit systems. The simulation framework in this study targets large-scale flexible transit systems with various vehicle operation schemes. Two transit systems are simulated, Real-time Shuttle service and High Coverage Point-to-Point Transit (HCPPT) in (16, 17) in a network of Orange County area. These two applications will show both the viability of simulation modeling as well as the importance of appropriately designing the simulation framework.

The paper is organized as follows. First, general concepts and special techniques related to modeling flexible transit systems are provided. Next, input data to construct the simulation framework are presented
including how the data are prepared and converted from the existing sources such as digital maps and trip demands. Then, the functional components corresponding to the proposed vehicle routing algorithms are explained as well as the fundamental simulation structure to comprise route planning. Finally, two applications are introduced and detailed simulation results are provided for analyzing the performance of operational schemes for flexible transit systems and for highlighting the benefits of the proposed simulation framework.

2. THE SIMULATION FRAMEWORK CONCEPTS
The proposed modeling scheme is for simulating any type of large-scale real-time routed transit (RTRT) service. This implies that operational schemes used in the proposed simulation should be able to incorporate any type of decision rules to dispatch vehicles to assigned passengers in a real-time manner. For instance, when simulating a real-time transit system, there are several requirements: (1) Vehicles need to be able to change their route at any time and to pick up and deliver passengers; (2) The network used in the simulation model not only contains the decision nodes and links for modeling realistic transit routing, but also covers enough area to evaluate the impact of network topologies for large-scale systems; (3) Solution algorithms are easily switchable to compare the system performance under various operating assumptions; (4) Individual objects in the simulation are tracked, such as vehicles’ location, passengers origins and destinations, and network performance at each simulation step; (5) All simulation elements are visualized to validate the simulation model properly, preventing programmer/user mistakes; (6) The simulation framework provides scalability regarding the number of vehicles, demand levels, and network scale because the computational effort in network algorithms are known to often increase non-linearly as the size of network increases.

3. INPUT DATA CONVERSION
The input data considered in this study cover a larger area than other microscopic simulation models. Unlike microscopic traffic simulation models that usually require all the implementation procedures such as network coding, network fine-tuning, parameter calibration, and OD estimation (18, 19), the proposed concept simply utilizes existing data sets for simulation network and passenger demand generation by simply converting data. It is noted that using such a concept might limit the simulation accuracy in terms of traffic dynamics due to the lack of vehicular interactions (e.g., car following and lane changing). However, the real benefit of this concept is flexibility and scalability for the real-world network topology especially when the focus of the study is aimed at feasibility and real implications.

The purpose of this section is to explain the data conversion process in which data from existing maps are converted to the simulation input format in Figure 1. It is assumed that a suitable source network exists such as a traffic network of a regional transportation model or a commercial GIS data. The simulator format has several data-type requirements, namely the transportation network, passenger demands, transit data, and user supply information as shown in Figure 1. Transit data contains transit facility locations and corridor information for a certain type of transit service. All simulation scenarios and parameters can be defined by a user supplied configuration format.

In this study, data from the SCAG (Southern California Association of Governments) transportation network and the OCTAM (Orange County Traffic Analysis Model) trip demand are illustrated as an example. Note that in the proposed simulation framework, the demand points do not necessarily match the corresponding network such as link or node because we assume point-to-point demands from any point over the simulation area, which can be generated independently without considering the transportation network. The simulation inputs can be created by the Main Converter (refer to the center block of Figure 1). Grayed data boxes are input files for the simulation framework.
Network Conversion Procedure

The transportation network contains key elements such as nodes, links, roadway attributes, and turning prohibitions for vehicle route planning. For example, the link and node layers in the SCAG network are in ESRI SHAPE files and those layers are converted to the road network in the simulation framework because the inherent data structure of SHAPE is for GIS analysis and data exchange, not vehicle route planning. The example SCAG network mainly covers Los Angeles and Orange counties. The data for intersection turning movement are contained in a binary file that is based on the network topology in the source network. It is worthwhile to point out that the main feature of the proposed simulation framework is that the input data can be replaced by any type of map, such as commercial digital maps, without excessive additional effort, as long as the data format is known.

If the source map comes from a regional transportation planning model, it might include centroids and centroid connectors. These centroid connectors are virtual links connecting centroids to real roadways, where the traffic is generated during the assignment procedure of transportation planning models. The centroid connects are obviously not part of the real network and no vehicles can move on them. Thus it is necessary to delete centroids and connectors during the map conversion process in Figure 2. It shows three stages: (1) Removing centroids and connectors in P1; (2) Converting node and link attributes to the internal data format in P2; (3) Validating network topology in P3. It is also important to note that when eliminating the centroids and the connectors, the node objects connected to the connectors should be modified because the centroid connectors are generally linked to intersections.
A procedure for validating network topology is necessary because logical errors or software glitches can occur. It should be able to support some functionality that can check properties of the converted map objects in order to prevent such problems. The proposed map converter supports two additional functions to investigate the potential errors in logic or programming: (1) Supporting interfaces to list the properties given the map object; (2) A route planning module that enables the user to check the network connectivity and link performance for vehicle path planning in Figure 2.

**FIGURE 2 Example of Network Conversion Process.**

**Physical Storage Format (PSF)**

The binary map format consisting of multiple tile maps covering the simulation area is designed to store geographical information. Sub-maps are created in the main map file which point to the location of each tile map and are accessed from the map header. Each tile map, called a parcel map, could have a different size of payload data depending on how many objects are included in the tile. Each parcel map has four tables with nodes, turning information, links, and link shapes. The objects in the node and link tables keep basic fixed-sized attribute so that those can be randomly accessed by object ID when the Data Access Library (DAL) reads the data. However, turning information and link shapes are stored in different tables due to the variable size of its records. These are not accessed by object ID, but offset bytes are given to access them directly from the objects in the node or link tables in Figure 3.
The concept of a parcel-based map design comes from the map data formats commonly used in commercial in-vehicle navigation systems. It is very efficient for vehicle routing and map display, especially when the coverage is large and the source map has a dense network. The concept is to divide the coverage area into many pieces of map tiles. Since fast object search is necessary for real-time vehicle tracking and identification of locations for trip requests, searching for the nearest object (e.g., links and nodes) based on geo-location data in the single large unit map would require intense computation. Alternatively, for the small-sized multiple tile-based maps, every parcel has area information and finding a nearest object can be done by searching through at most four parcels. Suppose that one large source map has $L$ number of links spread uniformly over the map. To find the nearest link from an arbitrary point would require $L$ comparisons without any geospatial index. Alternatively, only a maximum of $4L/n$ comparisons would be needed if the tiles were divided into $n$ units uniformly over the same map ($n > 4$) with an assumption that every parcel contains its area information. Moreover, the global coordinates can be stored in the form of local coordinates without loss of geo-coding accuracy. Another advantage of using the parcel approach is that object rendering is much faster because it reads only a few parcels’ viewpoints without spending much memory on reading a huge size of map data. The size of a parcel can be determined by the coverage of the source map. For instance, if the source map covers a statewide level network with detailed local streets such as that of a state like California, the parcel-based map is the better approach. However, the drawback of the parcel-based map design is that extra nodes must be created to connect two links that would have been one link in the large single map, but separated by two parcels. In summary, there are some advantages to using parcel maps including fast local search, fast display, and lower memory consumption. However dividing a map into many parcels requires keeping the connection information of links between two separate tiles.
Stochastic Zonal Demand Generation

As opposed to other simulation models such as conventional planning models and microscopic traffic simulation models in which trips are generated from the zone centroids, the simulation model for flexible transit systems deals with point-to-point trips where passengers’ origin and destination locations can be at any location in the simulation area, as minor streets are typically not in most traffic simulation networks. The trip demand data illustrated with OCTA model contain polygon-based spatial and temporal trip data in order to generate shared-ride trip requests. In order to generate such point-to-point trips, TAZs (Traffic Analysis Zones) in OCTAM are used, which is defined as a single polygon that usually represents the population and employment density to generate or attract trips in conventional regional planning models. A total of 1,220 TAZs are selected among the original 2,912 TAZs in OCTAM. The trip table contains passenger demand rates from one zone to another zone. The point-to-point trip demands can be randomly generated under the usual assumption of spatial uniformity of demand within the TAZ as per the trip table destination probabilities in each small TAZ. Once passenger groups are generated with a random location, they are registered in a data structure containing all their features and statistics, such as origin point, destination point, desired service time windows, number of passengers in group, and time stamp for each passenger event. Note that this allows for modeling even passengers from different households joining together and waiting as a group at a random point for a pick-up by a flexible-transit vehicle—the kind of possibilities that are not incorporated even in the few existing simulation software that generate multiple passengers for vehicles. Those registered passenger requests are called in real time manner during the simulation hours so that the vehicle dispatch algorithm does not have any information regarding the upcoming passenger requests. Alternatively, the simulation can import individual trip requests that are previously generated, which could be useful when applying historical based real datasets.

Terminal Facility and Trunk Network Design

Some flexible transit systems involve fixed service lines (trunk corridors) or passenger transfer facilities (hubs) as part of their service such as Mobility Allowance Shuttle Transit (MAST) in (20) or HCPPT in (16, 17). In such a case, service vehicles remain restricted on the designated links when they are moving and stopping. Hubs are available access points for passenger boarding and alighting, where many service vehicles frequently enter and leave. Concentrating the service vehicles on a certain site might cause congestion effects due to interactions with other service vehicles. However, such terminal operations are not simulated in the propose simulation. Instead, it is suggested that predefined congestion penalty parameters can be applied to both average vehicle speeds and passenger boarding and alighting times. Those predefined trunk corridors, hub locations, and penalty parameters can be provided by users as an additional input file. Once the file containing the hub locations and connectors is imported into the simulator, the predefined route information can be translated automatically by the shortest path algorithm using link performance during the simulation initialization.

4. SIMULATION FRAMEWORK

Simulation Objects and Events

Object-oriented simulation can be a useful approach in transit simulation. Each service vehicle can be represented as an independent simulation object composed of various entities interacting with the main dispatch center and passengers. Fleet operations’ event messages, such as move, stop, pick-up, and drop-off, are delivered by the main event handler. Once an event message arrives at a vehicle’s event message queue, the vehicle object can access a global object database, called the Common database, to get detailed vehicle schedules and passenger events. The dispatch center object is the main component that updates the Common database via receiving all passenger requests and all vehicles’ status in real-time. Passenger objects are randomly generated in the given temporal and spatial distribution. Unlike other simulation
objects, a memory pool is employed in object allocation since passenger objects are continuously created (show-up with pickup requests) and deleted (arrival at their destination).

### Main Architecture

The key challenge of the proposed framework is to simulate real-time flexible transit systems on a transportation network without any dependency on commercial simulators. The main elements of the simulator are the Simulation Container, the Simulation Core containing simulation objects, the Data Access Library (DAL) to get attributes of input data, the Vehicle Scheduler, and the Routing Algorithm for fleet operations. This process is depicted in Figure 4. We used an object-oriented programming (OOP) design scheme, which is comprised of a set of objects to represent the behaviors of simulation elements.

![Simulation Framework Diagram](image)

**FIGURE 4 Simulation Framework.**

The Simulation Container is a main component to read input files, create DAL, and the simulation core object according to the parameters defined in the configuration file. It also creates the graphic user interface (GUI) that allows users to track simulation objects such as vehicles’ schedules, locations of passenger requests, and waiting queues at hubs during the simulation run. Note that the simulation core is separated as an independent thread so that any unexpected interrupt from the GUI is prevented. The simulation process is synchronized by a timer that controls passenger request generation and vehicle position update. Once the simulation is started, the timer invokes passenger events for trip requests and the passenger events are kept in the system queue. When a new request arrives, the vehicle scheduler, which is equivalent to the dispatch center, is called to assign the new passenger and it updates the existing vehicles’ schedules by solving a vehicle routing problem. At the same time, the timer sends the notification for vehicles to move along their current routes based on given travel speeds of links. The speeds can be updated by auxiliary link travel time information, but generally a link look-up table needs to be prepared to translate the travel time information to the simulation network link levels, and then the vehicle speeds on links are updated by random variation based on the link speed. It is also not difficult to incorporate link travel time updates coming from an independent road traffic flow simulator module, or to update the travel times periodically with a synchronously running commercial traffic simulator, though this was not attempted in the simulation studies for which we provide results in this paper.
Vehicle Routing and Scheduling Core module is the key component that contains detailed vehicle operation rules and associated routing constraints as per user’s configuration, such as service zones, operation parameters, vehicle routing scheme, service constraints. It captures vehicle and passenger data from the simulated objects and creates a solver instance for vehicle routing problem. Various types of solvers can be connected for different transit applications accordingly. For example, HCPPT vehicle routing problem can involve Genetic Algorithm, Insertion Heuristic, and exact method with ILOG CPLEX and the Scheduling Core is capable of dynamically selecting different vehicle routing algorithms during a simulation run.

As the vehicle routing algorithm solves an optimization problem periodically, it is necessary to prepare the matrix for the expected travel times and distances for all OD pairs associated with both the current vehicle locations and the demand points. For fast route planning, we set up an additional pre-calculated database to store travel times and distances between major intersections. The pre-calculated database can be either based on the static information or based on the auxiliary time-dependent traffic information. As the system reconstructs the pre-calculated database, an optimal path and schedules can be changed. Notice that the link travel speeds used for simulating the vehicle movement are differentiated from the skim for vehicle routing and schedule so that the system can simulate the impact caused by unexpected traffic congestion or vehicle breakdown.

Passenger Generation and Door-to-Door Vehicle Routing

As shown in Figure 1, the Trip Request Pool (Demand data in Figure 4) stores the locations of passengers’ origins and destinations, which can either be prepared in advance as an input file or randomly generated with minimum information. Those origin and destination points are characterized as Longitude and Latitude (e.g., World Geodetic System 1984). When a new pick-up request arrives, DAL helps the main scheduler identify the passenger pickup point, consisting of the parcel map ID (the identifier of the unit map), the link ID (the identifier of the link), the moving direction (in forward or backward), and the position (of the link in meter from the upstream node) by projecting the origin point on the nearest link on the map. A binary indicator is used to identify a vehicle moving direction (forward and backward) based on link’s start and end nodes. This process is depicted in Figure 5.
The vehicle path is calculated by a link-based Dijkstra’s algorithm supporting door-to-door path-finding. Unlike the conventional node-based shortest path algorithm in which a searching point should be a node, the start and end points can be any points on the links or nodes. Also, it is known that link-based shortest path algorithms make possible to reflect vehicle turns effectively as studied in (21). The algorithm finds the shortest path by fixing link-labels instead of node-labels, which is essential for considering turning-prohibitions at the intersections in the network, when link-to-link costs have to be included in the path costs during the label-update process in the algorithm. Vehicle dwell time at passenger stops can be an important calibration factor, which represents passenger boarding and alighting times. The simulation framework assumes a random uniform distribution for service times. It is noted that such assumption can involve factors determining service times, such as internal layout of vehicles, number of doors, and passenger age.

Simulation Software (User Interface)

Another challenge of the proposed framework is to implement a comprehensive user interface, in which users can set up a detailed simulation configuration and control the simulation procedure. The Microsoft Visual C++ environment on Windows is used to build the simulator and to provide efficient GUI design. Similar to microscopic traffic simulation, vehicle movement and operation is observed continuously over time based on a fixed small time step. In our simulator, a one-second time step is considered under the assumption that the dispatch system tracks the vehicles’ positions and passenger requests second-by-second. Five main user interfaces of the simulator are illustrated in Figure 6. The object layers display: (a) Vehicle positions and events; (2) Trunk network; (3) Road network; and (4) Demand zones. A double-buffering technique is applied to minimize the resources of the Graphic Device Interface (GDI) objects when refreshing graphics with simulation elements. The system prepares two separate off-screen image buffers for static simulation elements (the first off-screen image with zones and network) and dynamic simulation elements (the second off-screen image with moving objects at every step such as vehicles, event identifier, and passengers), after which the system only needs to update the dynamic elements at every second of simulation. When the system refreshes GDI graphics, it merges both off-screen images and copies to the primary screen. The main benefits of these techniques are to eliminate the flickering problem that can occur in direct rendering and to save computational resources.

FIGURE 6 Real-time Flexible Transit Simulator: Object Layers and User Interface
In Figure 6, users can control the start/end/pause/speed/batch of the simulation. The property window reports detailed information of simulation settings, individual vehicle’s schedule, and passenger transfers at hubs. It is important to note that tracking simulation objects in the display and property window help users validate the developed simulation algorithms and designed operation schemes.

5. SIMULATION APPLICATION

HCPPT System

HCPPT is a new type of real-time transit system firstly proposed by Cortés and Jayakrishnan at University of California, Irvine in (16). The inherent design of the HCPPT scheme allows the modeler to decompose the large-scale system into smaller pieces of any shape (e.g., zones), and to formulate local optimization sub-problems that are apparently mutually independent of each other. High coverage in its name implies that the system is meant for real-time operations with a large fleet size. Assuming that unknown point-to-point travel demands are generated in real-time and large numbers of vehicles are dynamically dispatched to pick up and deliver passengers, vehicles take real-time optimal routes with travel on higher-speed arterial and highway (trunk) routes along with pickup and delivery of passengers in slower local streets (local). Each transit vehicle is assigned to a home area. When the vehicle travels out of the home area, it can only visit hub terminals in neighboring areas along trunk routes.

![Diagram of HCPPT System](image)

**FIGURE 7** (a) High Coverage Point-to-Point Transit and (b) Real-time Shuttle System.

Figure 7(a) provides an illustrative example. The service area is divided into many zones, and each zone has a predefined number of cells (e.g., seven hexagonal cells) with one hub location where passenger transfer occur. When a new passenger pickup request comes in, the best available vehicle in pick-up/delivery operation in the zone picks up the new customer and goes to the hub of the cell-cluster. Each vehicle has a re-routable portion within its home zones, but vehicles cannot be re-routed while moving on the trunk network to a given neighboring hub to which they are already assigned. For example, a vehicle that picks up a passenger traveling from A+ to A− can be re-routed before it gets the trunk network to go to Hub 4. The vehicle picks up all passengers who go to zone 4 and drops off the passengers at Hub 4, and then the passengers will transfer to get to their final destination by using another vehicle designated to zone 4. It means that one of the returning vehicles from Hub 1, 2, or 3 to zone 4 picks up the passengers at Hub 4 and delivers them to the destination. Therefore, travelers will at most transfer once. It is noted that passenger transfers could occur either at the origin hub or the destination hub of the passenger depending on the number of customers who go to the same destination zone. Detailed examples of the heuristic rules
Jung and Jayakrishnan

are described in (16). For solution algorithm development, Cortés and Jayakrishnan proposed a
generalized concept for local vehicle dispatch rules based on real-time stochastic control schemes in (17).
For HCPPT trunk network, Pagés et al. expanded the trunk network problem to the Mass Transport
Vehicle Routing Problem (MTVRP) in (9) and Jung and Jayakrishnan proposed the multiple-hub path-
based routing to optimize vehicle routing through multiple hubs by relaxing the condition of the strict
HCPPT operation rule in (22).

8 Real-time Shuttle System

Real-time shuttle service is another design variation of flexible transit system. The basic concept of real-
time shuttle system is similar to real-time shared-taxi system shown in (23), but the service employs
larger-capacity vehicle fleets in a larger area. The major difference in functionalities from the
conventional shuttle service is that it can be classified as real-time routed transit (RTRT), particularly in
cases where the passenger demand is not known. Moreover, it doesn’t require a certain type of hub-and-
zonal design that necessarily causes some passengers transfers at hubs as in the case of HCPPT. Service
vehicles continuously pick up and drop off passengers as they move around the service area as shown in
Figure 7(b). The basic scheme is to allow multiple passengers for shared-ride on any vehicles based on
real-time service requests, but time windows (maximum waiting time and maximum detour time) are
applied to prevent excessive passenger detour. For example, a passenger traveling from A to A’ does not
have to stop by a hub terminal for transfer, but it could entail a long detour to the destination because of
not taking higher speed corridors (trunk network) such as urban freeways. This service concept would be
beneficial when passenger demands are widely distributed without any higher demand corridors and the
passenger trip lengths are relatively short. Notice that passengers can be rejected if those are not met with
service availability. The dispatching rules and system performance were studied with various algorithms
such as insertion heuristic and hybrid simulated annealing by Jung et al. in (23).

Simulation Scenarios

Simulation experiments were conducted with two applications stated in previous sections. Table 1
summarizes the simulation configuration. The first experiment involves HCPPT simulation composed of
18 zones and hubs. Local and trunk lines are both served by the same HCPPT vehicles. The other
experiment is simply conducted by simulating a real-time shuttle service covering the same area. For
simulation scenarios, a total of 600 service vehicles are assumed with randomized initial positions over
Orange County network (366 mi²). We consider only one type of vehicle – a 7-seater van. The total
simulation time is 4 hours including 30 min as a warm-up period. Eight different demand levels from D1
to D8 are assumed with 1.8, 3.6, 5.4, 7.2, 8.9, 10.6, 12.5, and 14.3 passengers/mi²-hr. A total of 3% of
requests could have two or more than one passengers in one customer group. It is noted that the same seed
number is used for generating real-time passenger requests. An average of one minute boarding and
alighting times are assumed for each passenger, which follows a random uniform distribution U(0.5, 1.5).
Due to lack of available link travel times covering the simulation network, we assume that service
vehicles can travel at 60-90% of the posted speeds on the network, depending on the road functional
classes. It is noted that waiting time and detour time constraints are not applied to HCPPT. Instead, the
vehicles in HCPPT zones periodically move between local areas and hub terminals based on the cyclic
sequence of vehicle states defined by Cortés and Jayakrishnan in (16).
6. SIMULATION RESULTS

As stated previously, for statistics and user traces, the statistics module stores all system performance data, algorithm running time, and generated passenger requests like vehicle distance traveled, passenger request incoming time, passenger pickup and drop-off time stamps, hub transfers, and vehicle load factors. The statistics module keeps the track of all objects in heap memory then writes output files after an end of simulation. The simulation results are provided in Figure 8. The objective of both applications is to minimize passenger travel times, so the waiting and in-vehicle times are critical factors to compare two service operations schemes. Figure 8(a) shows waiting time at home and hub for both Shuttle and HCPPT. Average wait times at home show less than 12 minutes in most cases, which are reasonable considering the both real-time responsive transit systems. Waiting times at hub reveals that passengers spent up to 18 minutes at hubs to wait for transfer vehicles, which is an intuitive result because the smaller-sized vehicles are used for higher-demand trunk lines over 18 hubs. But it is important to note that the waiting times at hubs are very sensitive to fleet operational scheme and HCPPT zone design (including hub locations). The passenger transfer rule can be one of key factors of HCPPT performance, but the optimal design of trunk network and hubs locations are not considered in this study. For in-vehicle travel times, passengers stayed longer time in HCPPT system because visiting hubs for transfer causes detour routes to their final destination.

Regarding passenger delivery, the two applications show very similar performance in Figure 8(c). The delivered passengers in HCPPT Hub indicate the numbers of passengers arrived at hubs for transfers during simulation hours. The numbers of delivered passengers with higher demand levels are not higher as much due to the limited size of available vehicles. Figure 8(d) shows the passenger door-to-door distances, which can be defined as a distance for driving a personal auto without a ride share. The Shuttle service tends to serve passengers whose travel distances are shorter as the demand level increases because the dispatch algorithm of Shuttle service has the ability to honor the maximum detour constraint that makes the solution algorithm reject unattractive passenger requests whereas HCPPT takes all passenger requests with a First-Come First-Served (FCFS) scheme. Notice that the difference in passenger door-to-door distance increases as demand increases because the Shuttle dispatch algorithm could have more opportunities to match passengers for shared-rides.
Figure 8(e) shows an interesting result comparing vehicle operation miles. At lower demand levels (from D1 to D4), HCPPT shows longer vehicle operation miles than those found in Shuttle service. This is an impact caused by the HCPPT operation rule to stop by hub terminals even with a fewer passengers while Shuttle vehicles directly move to passengers’ destinations. On the other hand, Shuttle service can show operational inefficiency if there is a higher passenger demand corridor that can be replaced with larger vehicle operations utilizing trunk lines and passenger transfers. This issue is addressed with vehicle load factors in Figure 8(f).

Figure 8(f) reports average vehicle loads, which indicates an average number of passengers in a vehicle during the vehicle operating hours. Clearly, it can be seen that the seats in Shuttle fleets are almost fully utilized (average 6 seats among 7-seat capacity) while HCPPT takes only half of its capacity (average 3.2 seats/vehicle) to deliver the similar amount of passengers. That is, HCPPT has more potential to improve its operational performance. As mentioned earlier, the results provided here are primarily to illustrate the capability and flexibility that the simulation framework provides. Furthermore, it is strongly noted that different simulation details can be measured with different parameters, zonal design, hub locations, operation strategy, routing rules, and passenger demand characteristics. The detailed analysis in this study indicates the main benefit of the comprehensive simulation framework to develop and evaluate real-time flexible transit systems.
FIGURE 8 Simulation Results: (a) Average Waiting Time Home, (b) Average In-vehicle Time, (c) Number of Delivered Passengers, (d) Passenger Door-to-Door Distance, (e) Vehicle Operation Miles, (f) Average Vehicle Load.

As the focus of the paper is on presenting the peculiar requirements in simulation models for flexible real-time routed transit systems, and presenting the design considerations through the description of a simulation framework, it is important to mention the computational performance of the model. The simulation of the above large network context in Orange County, CA, over a 4.5 hour period required computation times from as little as 4 minutes to as much as an hour, depending on the routing algorithms modeled therein. For example, the Insertion Heuristic algorithm showed the best computation time of 4 minutes and the Simulated Annealing algorithm for global routing optimization required about 50 minutes (Intel Core 2 Duo 2.60GHz with 4G RAM on Windows XP).
It is also noticeable that large-scale flexible transit simulations that include the modeling of vehicle-routing can be done much faster than real time, namely 60 times faster, at best, and 4 times faster, at worst. This indicates the potential for use of the presented simulation framework in real-time optimization and control frameworks as well.

In addition, it can be seen that the simulation times are significantly faster than found for commercial software simulating auto traffic for networks of the same size. In our experience with commercial simulators such as PARAMICS and TransModeler modeling traffic on the same network as above on the same desk top computers, this is typically between 3 times faster than real time to about as fast as real-time. As the transit modeling framework runs several times faster for most routing algorithms and at least twice as fast even in the worst case, it is also seen as not a bottleneck in synchronous simulation of traffic and transit, as alluded above.

7. CONCLUSIONS

An elaborate simulation model plays a central role in evaluation of dispatch rules and operation schemes in flexible transit applications, especially those which are related to real-time dispatch systems. The main objective of this study is to develop a simulation framework for flexible transit systems, which could be comparable to commercial software packages. Simulation framework design and data conversion procedure are discussed at a detailed level. The main benefit of the proposed simulation model is a flexible architecture to easily apply various types of flexible transit operation schemes in conjunction with different vehicle routing algorithms.

It should be addressed that the proposed simulator is developed for different purposes than the microscopic simulation framework developed by in Cortés et al. in (7), which was specifically for studying the effects of feedback control of the vehicle control schemes (mainly local vehicle routing) where the dynamically changing travel times from traffic congestion was an important aspect. However, in this study, the focus is more on modeling optimization algorithms and operational design flexibilities and thus microscopic traffic simulation or an interface to a microscopic traffic simulation was not necessary. Finally, it is hoped that this study has pointed out the details that need important considerations in a comprehensive transit simulation platform.
REFERENCES

1. PARAMICS. http://www.paramics-online.com
2. VISSIM. http://www.vissim.com
3. AIMSUN. http://www.aimsun.com
