Public Transp (2017) 9:411–435
DOI 10.1007/s12469-017-0158-y

CASE STUDY AND APPLICATION

Enabling large-scale transit microsimulation for disruption response support using the Nexus platform

Proof-of-concept case study of the Greater Toronto Area transit network

Siva Srikukenthiran[1] · **Amer Shalaby**[1]

Accepted: 3 March 2017 / Published online: 20 March 2017

Abstract In many large cities, public transit systems have been carrying an ever-increasing burden of commuters. In such systems, service disruptions can negatively impact system performance and transit users well after they are resolved. Currently, transit agencies handle these disruption episodes in an ad-hoc fashion, largely due to the lack of adequate analytical tools to aid in analyzing and selecting appropriate response strategies. This paper presents a proof-of-concept case study of the Greater Toronto transit network using Nexus, a new crowd dynamics and transit network simulation platform. Nexus enables detailed simulation of all transit system actors using a novel method of linking together established simulators of surface transit, fully separated rail transit, and stations. Transit users, as agents in the model, move between the different simulators and have their routes determined by an external dynamic routing module. The case study focuses on interfacing Nexus with a commercial pedestrian simulator, MassMotion, to allow for detailed crowd simulation at key stations, and illustrating how the platform could be used for disruption management. To this end, the impact of disruptions of various lengths was analyzed, and a simple response strategy was implemented to provide an example of how the system could be used to test mitigating strategies.

Keywords Crowd dynamics · Disruption management · Microsimulation · Public transit · Subway networks

✉ Siva Srikukenthiran
siva.srikukenthiran@utoronto.ca

Amer Shalaby
amer.shalaby@utoronto.ca

[1] Civil Engineering, University of Toronto, 35 St. George St, Toronto, ON, Canada

1 Introduction

Public transit authorities around the world are considering a multitude of investments and strategies to combat ever-growing congestion across transit networks and deteriorating quality of service. The day-to-day functioning of transit systems is greatly challenged by recurring service disruptions of varying magnitudes and occasional threats of major emergencies. This is especially true in mass-transit systems that are currently running at capacity with short headways (Toronto Transit Commission 2008). The selection, prioritization and optimization of transit improvement strategies and interventions require a thorough evaluation using a high-fidelity modelling system. Currently, however, transit agencies rely on ad-hoc methods when responding to sudden disruptions in mass-transit networks (Jespersen-Groth et al. 2009). This is largely because of the lack of analytical tools capable of modelling and analyzing the network-level impacts of response strategies, especially with performance sufficient to handle large-scale systems in a reasonable time frame. A high-fidelity modelling system usable for disruption management support needs to both model transit supply in detail while accounting for the behaviour of passengers at both global and local levels; however, among the existing widely-used methods of transit network modelling, no single approach meets these requirements.

Traditionally, researchers and transit practitioners treat individual stations and rail lines as isolated from the transit network when attempting to precisely simulate passenger and train movements for evaluating operational policies. Analysis of crowd flow impacts due to crowd control or structural changes at stations are, therefore, usually performed without directly accounting for how these changes might affect train service. Similarly, evaluation of modifications to train movement, for example from new signalling systems, do not consider the resulting implications on crowds within stations. This disconnect has originated both from traditional divides within transit agencies, where train operation and station management are handled by separate entities, and similar siloing in the research community, where pedestrian and train flows have not been modelled in an integrated framework. The pedestrian modelling field has largely focused on understanding the basics of crowd motion, while the two-way interaction between crowds and transit vehicles at stations has yet to attract significant attention. This type of isolated analysis, while much simpler and requiring significantly less effort, can lead to uncertain results. As a result, there remains a significant gap for a dynamic modelling system with the capability to handle, in detail, both transit supply and passenger behaviour, and the flexibility to handle unexpected changes to the transit network.

To begin to bridge this gap, the authors previously proposed a framework for a new method of constructing large-scale models of transit networks to enable the incorporation of crowd simulation (Srikukenthiran and Shalaby 2011). This framework called for a modular approach to network creation, linking together separate simulators of the three main components of transit networks (surface transit, separated rail transit and stations), and a specific focus on properly modelling interactions at the interfaces between components. Transit users, as

agents, move between simulation services and have their routes determined by a decision-making module. This structure would enable the use of the most appropriate simulation software to avoid any deficiencies in fully integrated simulation packages. As an added benefit, it permits simulation runs to occur across multiple computers. Such a method also allows for more feasible construction of large scale regional networks, making possible a piece-meal approach. An initial prototype was previously developed to illustrate the basic viability of this modular service-based approach, tested against two hypothetical small and mid-sized networks (Srikukenthiran and Shalaby 2012). This paper details this next phase through a case study of the transit networks of the Greater Toronto Area (GTA) built upon an updated version of the platform, now called Nexus.

2 Current state of network-level transit models

The modelling of large scale urban public transit systems that is required for disruption analysis has predominantly fallen into two approaches, either macroscopic or micro-simulation based. They range from very abstract network topology analysis to microsimulation models.

The most abstract type of analysis is found within the discipline of complex network analysis. These types of analysis are statistical in nature, taking into account network characteristics like the distribution of the degree of connectivity of nodes, clustering coefficients and network size in comparing networks and deducing how they might perform (Lee et al. 2008). Public transport networks are the main target of complex network analysis, with all found to be examples of small-world networks, networks with the property of low direct connectivity between most nodes, but a low number of hops to travel between them (Sienkiewicz and Holyst 2005). Studies using this methodology have been conducted around the world with a particular focus on subway networks (Angeloudis and Fisk 2006; Sienkiewicz and Holyst 2005; von Ferber et al. 2009; Lee et al. 2008). Most of these studies have focussed on the characterization of the network, but some have gone beyond to examine how these network characteristics lend to grading network resilience (Chen et al. 2007; Angeloudis and Fisk 2006). Nevertheless, while these methods allow for a computationally efficient first-pass examination, their analysis is limited to network structure and basic examination of passenger flows. The transient nature of disruptions, and both passenger and service response cannot be modelled using these approaches.

More sophisticated analysis necessitates more detailed methods; however, current methods have notable shortcomings. On the macro-modelling side (e.g. EMME, VISUM), an abstract (graph) representation of the network is used, with aggregate demand models used to model network flow in order to perform transit assignment. The dynamic nature of transit network actors is difficult to capture in such an approach, allowing for only analysis of permanent large scale changes. On the other extreme, traditional micro-simulation software like Aimsun, VISSIM or PARAMICS are at-their-core traffic simulators, and do a relatively poor job

modelling transit service and demand (Cortes et al. 2010), particularly rail and the behaviour of transit users, lacking the capability to perform transit assignment.

To address some of these shortcomings, in recent years, researchers have generally followed two approaches. An overview of some prior efforts is described by Cortes et al. (2010). In the first approach, new simulators are created or existing simulators are adapted to either specifically model movements or interactions that are not currently captured properly by existing simulators or to test behavioural models (Cortes et al. 2010; Cats 2011; Wahba and Shalaby 2011; Lämmel et al. 2016b). In the latter case, however, these simulators (e.g. BusMezzo, MILATRAS, MATSIM) have been mesoscopic in nature, using simplified models of transit vehicle and passenger movements, limiting their application to surface transit. As a result, in the second and more sophisticated type of approach, researchers have taken advantage of the application programming interfaces (API) of commercial or some open-source traffic simulators to incorporate new detailed models of transit service routing and passenger behaviour (Cortes et al. 2010; Rieser 2010; Su et al. 2015).

While this allows for an improvement over the base simulator, one is still confined to a single piece of software for modelling of all transit modes, and has to accept any performance shortcomings when simulating large-scale networks. Looking to improve scalability and not be confined by this limitation, Lämmel et al. turned to a similar approach as was previously proposed by the authors in Srikukenthiran and Shalaby (2011), namely allowing for simultaneous simulation of pedestrian and vehicles in separate software, possibly running on different computers. For the purpose of modelling evacuation of a relatively small population of less than 8000, MATSIM's configurable mesoscopic simulation system was used as a base, linked via a remoting protocol to JuPedSim, a microscopic pedestrian simulator for modelling movement within a single train station (Lämmel et al. 2016a).

In all of these prior efforts, however, the effect of crowds on transit performance has been ignored, critical for modelling of congested high-frequency metro networks. This, however, has recently begun to change with some leading network simulators incorporating pedestrian modelling within their software, often partnering with existing standalone software, to permit analysis of pedestrian-vehicle interaction. In addition, while the focus remains mainly on surface transit, some simulators, like PTVs VISSIM, have added some limited capability in simulating mass transit and the influence of pedestrians on dwell time during boarding and alighting. However, as these simulators are constrained by their auto-focussed origins, pedestrians do not make trips through the network but are introduced and removed after their walk through local spaces (PTV 2012). Pedestrians simulated in this fashion have no awareness of network changes or the trip they are making.

This leaves open potential for a system that can combine micro-simulation modelling of transit supply with advances in crowd dynamics and passenger behaviour modelling to address current limitations that prevent detailed analysis of complex issues in transit operations research and disruption management.

3 Distributed simulation systems

In adding detailed pedestrian movement to network simulators, performance can be sluggish, especially for larger crowds. While some progress has been made by adjusting how collision detection is performed (Still 2000), or harnessing multi-core processing in the case of the MassMotion simulation software (Patel and Hudson-Smith 2011), the simulation of large numbers of people cannot be accomplished on a single computer at speeds necessary for time-critical applications. In the more mature discipline of traffic simulation, however, the concept of spreading simulations over multiple computers has been attempted, either through the use of clustered computing or over a local area network (Liu et al. 2005; Klefstad et al. 2005). In either case, this requires partitioning of the network into cells, distributing the calculation of vehicle movement for each cell across computers, and having a mechanism to deal with transitions of vehicles as they cross cell boundaries (Liu et al. 2005; Klefstad et al. 2005). While this may be possible in traffic simulations where vehicle flow is relatively low and trajectories can be more easily predicted to aid in the transition process, it is problematic when simulating large crowds in 2D or 3D environments. As a result, a different approach is taken for the Nexus platform to simulate city-level transit networks.

4 The Nexus platform

To allow for modular programming, expandability and flexibility, Nexus is structured using a distributed service-oriented architecture, a server-client configuration that can be run either on a single computer or spread across multiple computers. Pedestrians, as agents, move between simulation services (station, train, surface vehicle), and have their routes determined by a transit assignment module. In this way, Nexus uses the natural divisions of the transit network to guide distribution of processing. Also, in order to be able to analyze movement at the network level, scalability is prioritized by incorporating the ability to handle models of varying methodology (detailed movement, queuing, etc.) simultaneously, communicating via a standardized interface. This method permits the interchanging of the software running each module without having to be concerned with other system components. The main driver of this structure is to allow coupling of otherwise separate simulation software (pedestrian, line, surface vehicle) in a coherent way to track the movement and experience of agents and the performance of the transit network.

The various components of the framework (Fig. 1) are defined by specific tasks. Individual simulation scenarios are specified and initiated by the network analyzer, which also processes output and allows for run-time visualization. The linked simulation components (coordinator, station, line, surface and pedestrian simulators, transit assignment) can be thought of as one service; it can be accessed to perform a simulation given a set of operational data input, producing detailed results as output logged to the database to be analyzed, and describing pedestrian and transit system

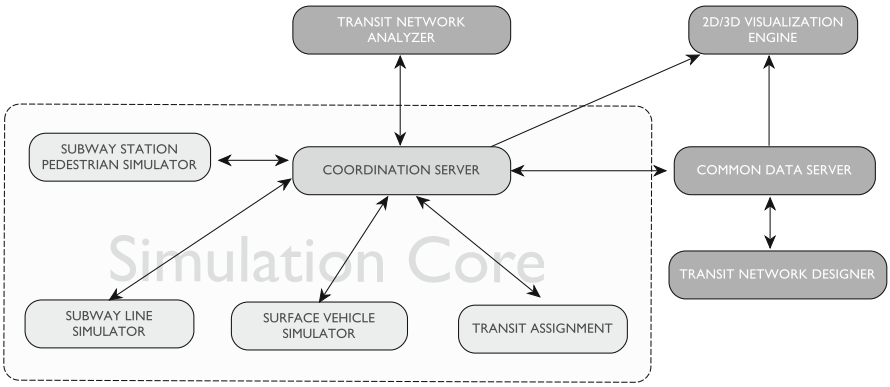

Fig. 1 Nexus platform components

level of service. The transit assignment module acts as the key decision making component, deciding on pedestrian paths through the network and allowing for rerouting in cases of information provision. The uniqueness of the Nexus framework is dependent on three features: (1) a hybrid simulation coordination engine to allow both time-step synchronization points, but time-of-event transfers of vehicles between simulation components, (2) a communication interface to allow components to send sensor feedback of network importance and allow for control signals to affect change on-the-fly, and (3) the ability to interface with existing commercial simulators to model the various transit components (stations, rail and surface operation).

Key to proper functionality is the mechanism of the Nexus controller, which is used to maintain proper synchronization between modules, while properly handling transitions of vehicles and pedestrians. To maintain synchronicity, it advances all simulators at regular intervals (at a period larger than the internal clock of any individual simulator) to allow for the modules to move forward relatively together. In addition, at these regular intervals, the transfer of low-volume pedestrian movement between stations and the street occurs, premised on these movements not having network significance and allowing for post-positioning without jeopardizing the integrity of the simulation. Large-scale movement between transit vehicles at stations, however, are deemed to be more time-critical, and as such, the controller allows for vehicle movements between modules (e.g. a train arriving at a station) in between synchronization pulses. Such a method allows for time-step and discrete event simulators to work together, without having to worry about rollbacks normally required in distributed simulation systems.

This structure used by Nexus is similar to the High Level Architecture (HLA) concept developed by the U.S. Department of Defence for defence and combat simulations, with some minimal application in traffic simulation (Kelin et al. 1998). HLA was devised to allow linkage of otherwise separate simulators together, providing infrastructure to handle interfacing, data exchange and time synchronization, and rules that must be followed to be compatible. The Nexus platform is structured in a way that has some common features with HLA, but in contrast to the

more general specification of HLA, constricts simulators to a few types, with clear geographic separation, and with custom interfaces built around their transit features as detailed above.

5 Prototype system

The Nexus platform prototype, with initial implementation described in Srikukenthiran and Shalaby (2012), was built leveraging the Microsoft .NET Windows Communication Foundation (WCF) to handle inter-application communication; WCF is a services and distributed computing architecture that allows for communication between clients and services running both on the same computer or spread across computers over a network. While the central software is written in C#, this does not restrict all components to this programming language, using the WCF web protocols to allow interfacing of software written in other languages and platforms. The focus of the initial implementation was to examine the feasibility of the approach at a structural level, with priority given to the computing architecture and the development of the Nexus controller. The individual modules (line, pedestrian and station simulators, and transit assignment method) were built as simplified constructs. It was shown that computation time could be decreased by distributing components across multiple computers without running into communication bottlenecks for large numbers of agents (Srikukenthiran and Shalaby 2012).

Before the case study was conducted on the GTA transit networks, several improvements were made to the original prototype to allow Nexus to handle a real-world large-scale transit network, and test disruption and response strategies. Vehicles were modified to be independent of a specific route, allowing for trip chaining across various routes. This was performed to bring Nexus in line both with how transit agencies would organize vehicle runs and to conform to the current standard in publicly available transit data, the Google Transit Feed Specification (GTFS), to allow for easier import of transit network data. Second, to provide Nexus with the ability to simulate detailed crowd movements in key stations, the commercial pedestrian simulator, MassMotion, was modified to interface with the system, allowing for runtime manipulation of agents and trains, and retrieval of pertinent info about conditions within the model. As part of this effort, additions were made to the MassMotion software to more realistically model the behaviour of agents at train platforms and when moving between station levels, by incorporating results of prior research efforts (Srikukenthiran and Shalaby 2015; Srikukenthiran et al. 2014). Finally, to enable simulation of train disruptions, the Nexus interface was modified to allow for two-way control of trains at stations, with the train dwell time set either by the line or station simulators. The final state of each of the major components are detailed in the following sections.

5.1 Coordination server

The Nexus coordinator acts as the main controller of the system with all communication between network modules occurring via this component. The

engine is specified via two separate interfaces. The first defines how clients (such as the network analyzer) interact with the engine, while the second is a call back interface necessary for data exchange with component services. While running the simulation, the engine requests from the transit assignment module the set of agents that depart from home during the current cycle, and pulses all component simulators to advance for the cycle duration and return sensor information when complete. Relevant information (e.g. demographics, walking speeds) can be attached to each agent to allow for uniform behaviour across components. The call back interface exposes functions to relay vehicle arrivals and departures (through transfers of complete vehicle data and their passengers), indicate expected vehicle arrival and departure times, update agent trip information and allow for general system logging. The vehicle-related functions are critical for components to be aware of vehicle arrivals or departures that might occur in the current simulation cycle, and to allow for pedestrians to be made aware of their movements. Agent trip information is updated by relaying pedestrians to the transit assignment module to retrieve the next trip leg anytime they move through doorways, board or alight. The engine handles all issues in determining which service component should receive pedestrians or vehicles who are transferring based on internal tables, allowing connected components to not concern themselves with how the transit network is distributed amongst simulation services and how transfers occur.

5.2 Transit assignment

The transit assignment module acts as the brain of the system by handling all decisions made by pedestrians at a network-level, particularly which routes to take to arrive at their destination at the appropriate time. An all-or-nothing schedule-based path-finding method was used in this implementation. A choice set of attractive paths was first determined by considering all transit paths with a maximum of two transfers (sufficient for 96% of trips in the case study network) between stops near to the origin and destination. Travel times were based on published schedules, with transfer penalties added; the quickest route was then chosen for each pedestrian. This simplified implementation is sufficient for a proof-of-concept model as detailed in this paper; however, an improved and calibrated assignment would be required for network-level validation of passenger flow and service performance.

In the final implementation, it is expected that MILATRAS (Microsimulation Learning-based Approach to Transit Assignment) will be used as the transit assignment engine. MILATRAS is a novel transit assignment method which uses a population of agents which learn to choose their path through the network and appropriate departure time using a learning-based procedure, converging over many iterations (Wahba and Shalaby 2005). It has been shown to perform comparatively well against traditional equilibrium-based transit assignment methods such as EMME, while allowing for policy-sensitive analysis (Wang et al. 2010). Incorporation of MILATRAS into the presented framework will allow for more accurate performance evaluation by incorporating crowd flow for use within its decision-making processes.

5.3 Stand-in surface, line and station simulators

As with the initial prototype (Srikukenthiran and Shalaby 2012), stand-in simplified simulators of surface, rail and most stations were put in place. As the Nexus system is more fully developed, these will gradually be replaced by more accurate software (either commercial or custom-made) in a similar fashion to the interfacing of MassMotion described in the following section.

The network structure and schedule contained within the Nexus database (based on publicly available transit data) were used to construct surface routes and generate vehicles for the surface simulator. This included auto generation of the paths along roadways taken by surface vehicles from path information provided in these data sets without specific modelling of the roadways. As the key goal of Nexus is to model large-scale mass transit and the impact of subway disruptions, while the surface network was incorporated to feed agents into the rail system, it was not the primary concern. As a result, a simplified mode of vehicle movement was used, with complete adherence to service schedules. Surface vehicles were advanced at an average speed based on their schedule times along their routes without modelling any other road traffic or intersections.

As was done with the surface simulator, the structure and vehicles of the rail network were also constructed via public data, and modelled in a simplified fashion. While train movements are generally governed by a complex interplay between track layouts (including segments and switches) and signal operation, for the purpose of this proof-of-concept, the rail model was constructed similar to surface transit, with vehicles moving along a single track between stations. After initial release based on their scheduled departure at the beginning of each trip, trains were set to move at an average speed based on their posted schedules. Two methods of schedule adherence were programmed, one where adherence was only maintained at the beginning of a service cycle and a second where trains were held where necessary to be kept on schedule at each station. The method employed was dependent on the transit service. Trains were modelled with their specific constitution, allowing for setting of the number of cars and capacities per car, dependent on the transit line; this information was passed along to the station simulators to use for boarding/alighting operations.

Finally, for stations where MassMotion models are not required, the station simulator also used a crude form. Agents were jumped between key points in a station (platforms, doorways) based on average adult walking speeds (1.3 m/s) without considering congestion. Two models of platform behaviour were programmed. The first (for commuter rail) was a simplified model of even distribution (agents assigned uniformly across the train doors). The second was a more sophisticated model previously developed for high frequency subway service, which simulated the spreading of agents using a diffusion-inspired mechanism as they entered a platform of specified length and width, and including the concept of preferred waiting locations based on the location of exits at their alighting station (Srikukenthiran and Shalaby 2015). Using the resulting distribution of passengers as an input (with capacity limited via setting a maximum local density), dwell times were set based on Westons formula (Harris and Anderson 2007). Agents were

allowed to board up to a time limit of 90 s at which point the doors would close; this value was arbitrary in the base model to avoid excessively long dwell times.

5.4 Interfacing MassMotion

MassMotion is an agent-based three-dimensional pedestrian simulator. The environment is built using 3D architecture, with separate floors connected with level-change elements like stairs, escalators, ramps and elevators. Agents enter and exit the environment via portals, and can be given a series of tasks to do in the meantime, including visiting specific areas and being involved in queueing processes. Agents can be assigned various profiles, modifying parameters related to body radii and walking speeds, as well varying sensitivity to direction of movement and waiting. As with most agent-based pedestrian models, movement in MassMotion is governed by two layers of guidance. The first, a short-range 'reflexive' model, adapts the Social Forces concept, with surrounding agents and obstacles producing forces on individual agents that guide their decision of velocity at each time step (Morrow 2011). This behaviour has been calibrated against Fruin's Level of Service (LOS) standard to ensure that speed-density profiles are consistent with field values (Morrow 2011). The second part is a longer range pathfinding model, based on an application of a modified Djisktra's algorithm on a network representation of the entire space, with a cost function that takes into account a variety of factors including level changes (Morrow 2011). This navigation model also considers congestion along the route, allowing for responsive agents.

In addition to basic way finding, MassMotion contains elements that allow for more complex behaviour. These include process chains, to simulate queuing processes as would be present at ticket counters, and gates, allowing for controlled flow and simulation of doorways. These tools allow for it to be used to model passenger movement through stations, with some limitations. Capacities of various sections, such as platforms, are determined automatically based on the 3D model. Portals provide entrance and exit locations at doorways and train cars, while links with timed gates permit the modelling of opening and closing of train doors. To simulate boarding and alighting processes, the doors are kept open until either no agents are within 2 m of a car door or the car hits capacity.

Interfacing of this software to act as the simulator of train stations was a key goal of the prototype, to enable much more detailed and accurate modelling of pedestrian movements. As the first commercial piece of software to be linked to the Nexus platform, the mechanism of interoperability between Nexus and any commercial software had to be devised. The commercial simulator does not interact directly with the Nexus platform, instead being linked via an intermediary wrapper application. This wrapper application is hosted as a service to conform with the service-oriented architecture of Nexus, exposing specific functions required for the station simulator component; these include the ability to load project files, control simulation, introduce agents and retrieve agent trip information, send and receive vehicles, and provide data on station performance. In general, it is expected that external commercial software have an application programming interface (API) to permit these functions. However, in the case of MassMotion, as this did not exist,

this interface was first developed, encapsulating MassMotion within a dynamic-link library; this included handling issues of programming language inter-op, allowing for communication between MassMotion written in C++ with the C# implementation of Nexus.

While the API allowed for dynamic manipulation of agents during simulation and retrieval of simulation metrics, the MassMotion model also had to undergo some minor modifications to make it properly interface with Nexus. These included removal of all agent schedules, renaming platforms to be consistent with the naming convention assumed by Nexus, and the addition of portals along the length of each platform to indicate waiting locations. Data was also included on how the identification numbers of transit stops (defined following GTFS processing) mapped to the portals of the MassMotion model. For the purpose of this initial prototype, only a single agent profile was used (the default commuter profile built into MassMotion).

5.5 Prototype system outputs

As a simulation platform that uses the capabilities of external simulators, Nexus is limited only by these other software in the metrics produceable by the simulation. To prevent information overload when viewing results, by default, only specific data is stored; this includes vehicle positions and passenger loads, vehicle arrival and departure times, total station counts, and trip logs of all agents (including travel and waiting times). In addition, each sub-component can specify, through registration of virtual sensors, specific metrics to report; this can include items such as platform counts as used in the case study below, but there is no limitation beyond what the individual software can track. Finally, all detailed simulation data produced by sub-components are also stored for viewing and analysis by their respective stand-alone software. For this prototype, for example, detailed MassMotion simulation output data is produced, allowing for the full range of analysis possible by MassMotion, including graphs of counts on various areas in the station, and density and flow maps.

6 Case study: goals

The case study model of the GTA transit network detailed for the remainder of this paper is presented purely as a proof-of-concept to illustrate the capabilities of the current version of Nexus. The focus is on detailing the data inputs required to set up the model in Nexus, demonstrate Nexus' technical ability to handle the simulation of a large scale real-world multi-modal transit network while interfacing with a commercial pedestrian simulation software for simulating key interchange stations, and illustrate how it could be used to test the impact of service breakdown and disruption response, both by the transit service and its users. Also discussed is the

unique piece-meal construction method, allowing for the model to be more easily improved over time, and the use of publicly available data sources to easily update routes and schedules. This allows for the same model to act as the proof-of-concept demonstration here, and later evolve to be a more accurate representation of the regional transit network as its individual model components are improved over time.

7 Case study: background on the Toronto Transit Network

The Toronto Transit (TTC) Network is Canada's largest transit network, serving over 1.6 million passengers daily, 900 thousand of whom use the subway network. Service is provided along 3 subway lines (Fig. 5), 1 intermediate capacity rapid transit line, 11 streetcar and 141 bus routes. It is also characterized by having an extremely high level of integration between its surface routes and subway network, with bus or streetcar service generally starting service at one of the 69 stations, and seamless transferring between the modes. Today, the subway network is known to be currently running at capacity, with issues of severe crowding at key interchange stations (Toronto Transit Commission 2008). As a result, it presented itself as an ideal real-world network for a system aimed at better taking into account these various interactions between adjacent modes and the effects of crowds.

The bus network in the Toronto transit network plays a primary role in providing feeder routes into the subway network. As agent trips are followed from origin to destination, this necessitated modelling of all surface routes in addition to the subway system. In addition, due to the high level of inflows into the City from surrounding regions, some method was required to introduce these flows into the Toronto network. To ensure Nexus' flexibility to handle multiple agencies and multi-agency trips, many of these flows were introduced by direct modelling of the transit systems of the surrounding municipalities and the GO train network.

With these additions, the final network consisted of several agencies, over a dozen rail lines and a hundred stations, and several hundred bus lines. The characteristics of each agency are shown in Table 1.

The period of simulation analysis was the weekday morning peak-period from 6 to 9 AM. To allow time for vehicles to enter the network and have an existing population of agents at 6AM, the simulation period was expanded to 5–9 AM.

Table 1 Characteristics of transit agencies modelled

Agency	# Surface routes	# Train routes	# Stations	# Stops
Toronto Transit Commission	177	4	69	10,922
Brampton Transit	44	0	0	2389
York Region Transit	133	0	0	4659
MiWay (Mississauga)	96	0	0	3065
GO Transit	0 (omitted)	7	63	268

8 Case study: data sources

8.1 Transit network structure and schedules

The Nexus data model was designed to be compatible with the GTFS, to allow for easy updating of route and schedule data. As a result, the data used to build the transit network structure and service information for the case study was mainly sourced from public GTFS files provided by the transit agencies in the GTA region. As the process of distributing public GTFS data is still a relatively new concept in the region, limitations existed in being able to acquire GTFS data from the several agencies in the region corresponding with the same time period. While this was not ideal, as a calibrated model was not a specific goal for of this study, these time disparities were not believed to be a critical issue.

While GTFS files are the standard method of input of transit data into Google Maps, the standard is not strict. As a result, the conversion of the GTFS data into a format suitable for use by Nexus was not a direct or easy effort, because of gaps in data or inconsistency in naming or numbering conventions, even within the same data set.

To convert the data within the GTFS files into a format compatible with Nexus, a program was written that could accomplish this goal. This included exploiting patterns in naming convention to extract station names and automatically categorize stops into platforms at stations or street stops. The occasional gaps or errors in data were reconstructed where possible or taken from other sources where information could not be automatically filled in (such as missing route path data for the GO train network).

In addition to this data, two additional sets of information were manually collected: the locations (longitude and latitude) of all station doorways, and the positions and associated doorways of all platform entrances and exits for the TTC subway network.

8.2 Agent origins, destinations and departure times

The Nexus prototype takes, as input, three items at minimum in order to generate a set of agents: the origin and destination of the agent trip and the departure time. Other information (e.g. demographics, walking speeds, other preferences) can also be attached for use by simulator components; however, for the purpose of this proof-of-concept study, this data was not input. For this case study, the required information was sourced from the 2011/2012 Transportation Tomorrow Survey (TTS).

The Transportation Tomorrow Survey is a wide ranging traveller behaviour survey conducted every five years in the Greater Toronto and Hamilton region. The survey is one of the largest of its kind in North America, randomly sampling 5% of the population in the region. The resulting data contains both detailed demographic information as well as a travel itinerary for a typical weekday. A section of the survey is dedicated to transit users, collecting data including the routes taken, access

and egress modes, and boarding and alighting subway stations. Also collected is the type of trip, describing the origin and departure locations (e.g. home-based work trip). In presenting the data, the region is divided up into over 3500 zones, with their size based on density.

For the purpose of this case study, information was extracted that provided the surveyed number of transit users (scaled to a full population) that commuted from each origin zone to each destination zone, in departure time divisions of 10 min. The population of agents (numbering 492,000) were then synthesized by distributing them over regions of appropriate land-use type. For example, origins beginning at home were placed in residential areas, while office destinations were placed in commercial zones.

The final distribution of the origin and destination points created for the entire morning period are shown in Figs. 2 and 3. As expected, trip origin points are much more widely distributed, with trip destinations concentrated in the central city and other known growth hubs. Figure 4 shows the top 100 desire lines between origins and destinations, further illustrating the concentration of commutes towards the downtown core.

9 Case study: building the base simulator network

The case study network was divided into 1 street simulator simulating all surface routes for all agencies, 2 line simulators (one for the TTC subway network, and one for the GO commuter rail network), 69 station simulators for the TTC network, and

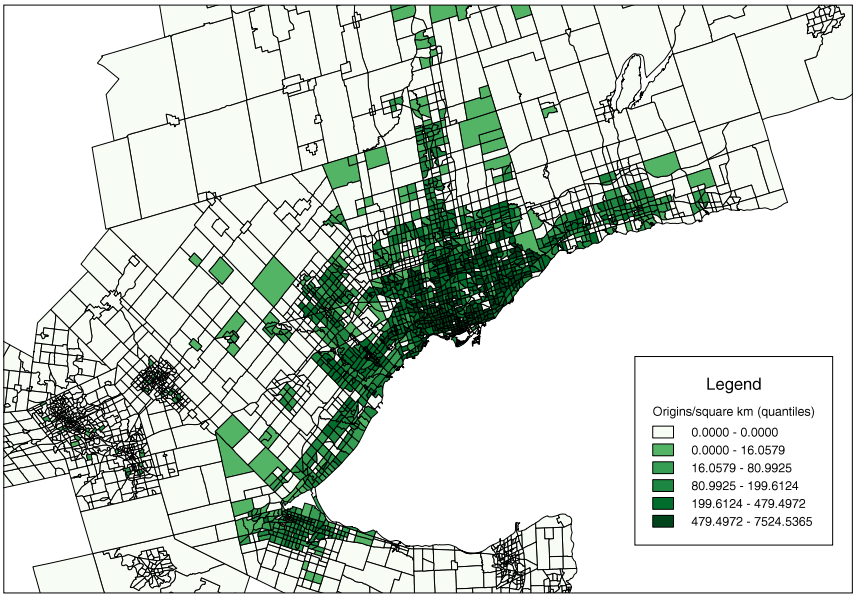

Fig. 2 Distribution of origin points (traffic zone level) for the case study

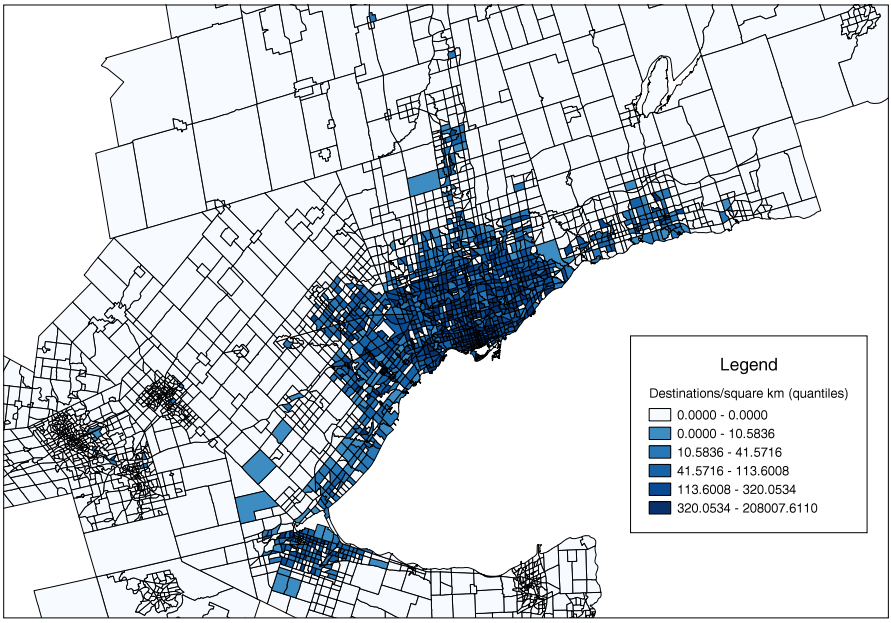

Fig. 3 Distribution of destination points (traffic zone level) for the case study

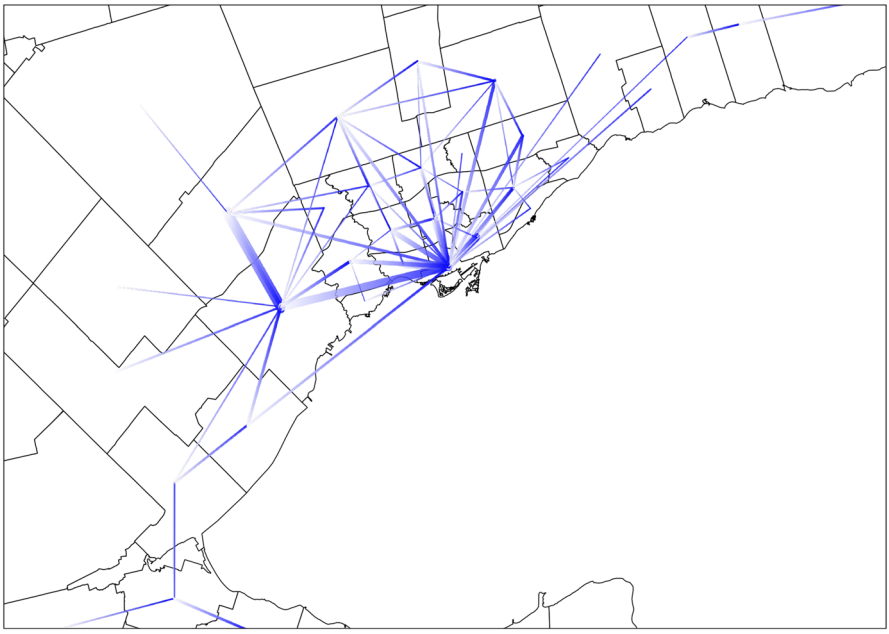

Fig. 4 Top 100 desire lines (planning district level) for the case study (direction towards darker gradient)

63 station simulators for the GO network. Except for two key stations, as detailed in the next section, all models utilized the simplified simulators described earlier in the paper for surface transit, rail transit and subway stations.

9.1 MassMotion models of Bloor/Yonge and St. George stations

Two key stations within the subway network were constructed within MassMotion, Bloor/Yonge and St. George. Their locations (shown in Fig. 5) are at the two main transfer points between the horizontal Bloor-Danforth line, and the U-shaped Yonge-University-Spadina line; as a result, they are the two busiest stations in the TTC. Bloor/Yonge station, in particular, is well-known to be over-capacity during peak periods, and is of key interest to the TTC with respect to better crowd management (Toronto Transit Commission 2008).

The model of Bloor/Yonge station was adapted from a prior model built to conduct analysis on how train arrival patterns would affect station flow. Details on its specific method of construction and validation can be found in King et al. (2014). Trains were added as a series of floors (one for each car), and four gated links to represent the doors (Fig. 6). At this stage, detail at the concourse level was minimal, with the ticket booth area and turnstiles not specifically modelled. This was not deemed to be particularly important in the absence of a higher fidelity model of the surface network (where pedestrian movement was greatly simplified); the focus in this study was, therefore, on proper modelling of transferring agents.

The use of the Bloor/Yonge model within the framework was key both as being a critical station in the network, but also provided a demonstration of the ability of the platform to enable easy re-purpose of existing models for broader network analysis.

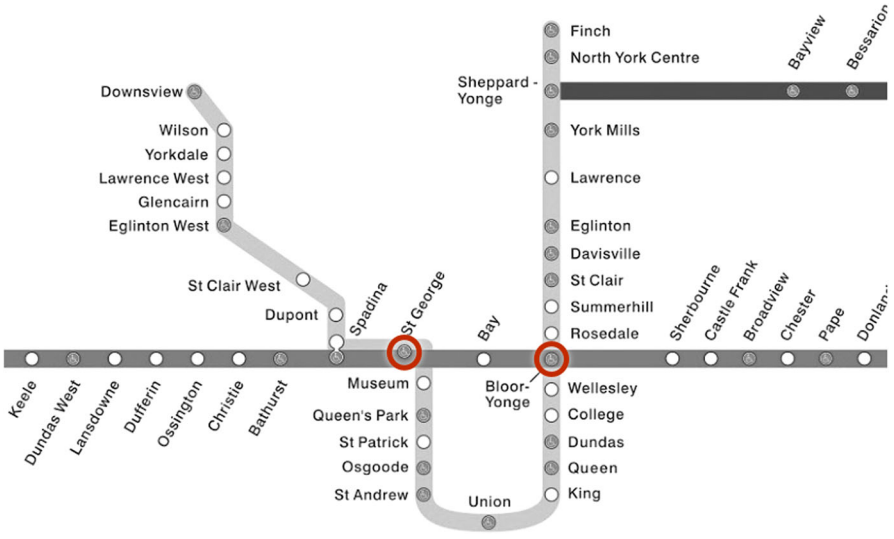

Fig. 5 Schematic of the central section of the TTC subway network highlighting St. George and Bloor/Yonge stations

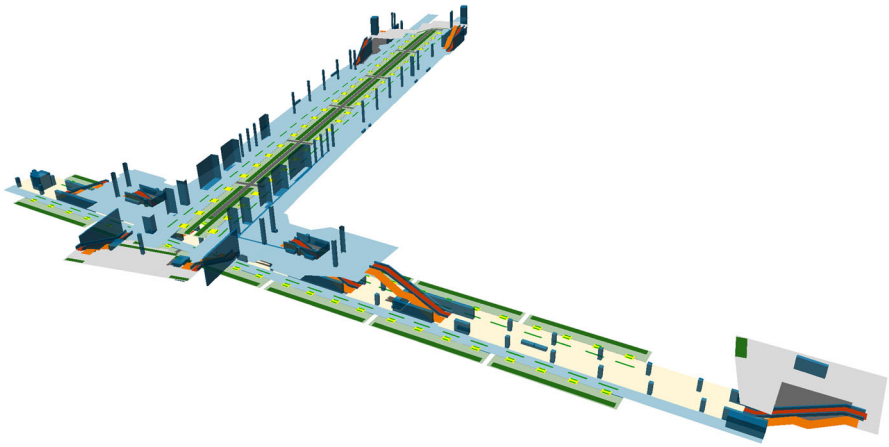

Fig. 6 MassMotion model of Bloor/Yonge Station

To maintain fidelity, the previously calibrated parameters of agent behaviour were maintained when adapting the model for use in the network.

The model of St. George station was constructed using provided floor plans. As no data collection was performed for this station, specific calibration of model agent and routing parameters were not conducted beyond minor modifications to ensure that unnatural agent flow did not occur (e.g. agents becoming stuck when walking in opposite directions on the narrow platforms). Figure 7 presents the final model, showing the station's four levels including the two train platform levels (for Bloor-Danforth at the lowest level, and for Yonge-University-Spadina above it), and single bus platform at the street level.

9.2 Operational parameters

In addition to the physical structure of the test case network, several components included variable processes and parameters that could influence network operation and agent behaviour. Their values predominantly defined the base model situation

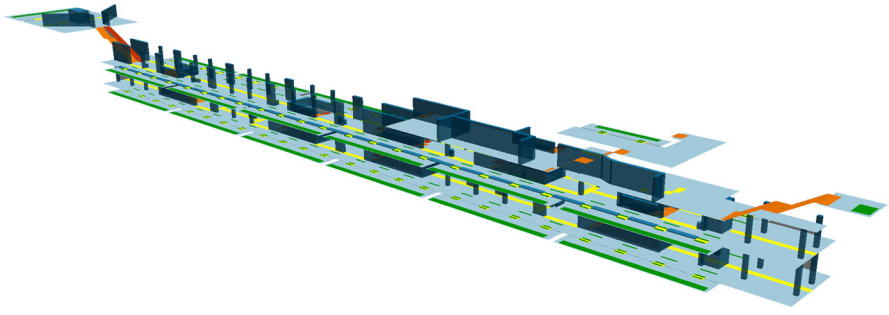

Fig. 7 MassMotion model of St. George Station

of the case study. While default values and operation, as described in Sect. 7, were used for most components, some case-study specific parameters were necessary, as described below.

For all of the TTC subway stations modelled in a simplified fashion (i.e not in MassMotion), the aforementioned diffusion model was implemented (Srikukenthiran and Shalaby 2015); this model was adapted for MassMotion by having agents target specific waiting portals. For GO Train platforms, as the service is relatively infrequent (>20 min) and run based on a schedule, an assumption was made that users would be more likely to distribute themselves evenly, independent of platform layout. As a result, the second platform model (uniform distribution) when boarding was assumed.

Subway trains on the TTC used the first model of schedule adherence, only ensuring trains adhered to their departure time at the terminal stations at the beginning of their trips. On the other hand, vehicles on the GO commuter rail network were held at each station until their scheduled departure time, to conform with how GO runs their train service. Finally, in both networks, only one vehicle was permitted to occupy a platform at any given time. The capacities of each vehicle were taken from published specifications for each type of vehicle. The TTC train network has three configurations of vehicles. Trains on the main Bloor-Danforth and Yonge-University-Spadina lines consist of 6 transit units, each with a capacity of 167 individuals. The Sheppard line uses the same car-size, but with only 4 transit units per train. Finally, the intermediate rapid transit, Scarborough line, have 4-car trains, each able to hold 54 individuals. For the GO network, as a detailed breakdown of the structure of trains on each line was not available, an assumption was made that all trains were at their maximum length and capacity (12 carriage trains, with each carriage able to hold 162 agents).

10 Case study: analyzing the impact of disruptions

To illustrate proof-of-concept of the use of Nexus for disruption management, a simple scenario was devised. To simulate the situation of a passenger pressing an assistance alarm, a hold command of various durations was sent to a train at a busy subway platform at the beginning of a high load period, without modifying the behaviour of all other service vehicle and transit user actors. As this type of disruption solely involved train movement, the signal necessary to simulate such an event was sent to the line simulator, overriding the normal situation where the station simulator would determine the dwell time based on boarding/alighting operations. It was implemented by extending the dwell time (by the specified duration) of the first train that reached the platform after the specified time of disruption.

The disruption was simulated by holding the first train to reach the southbound platform at Bloor/Yonge station after 7:55 AM, just before a rush hour surge of passengers. This platform was selected given its high volume and overall critical importance to the subway network. Disruption lengths were set at 10, 20 and 30 min

to gauge the ability of the modelled subway network to recover to normal crowding levels without any change to the operation of other trains.

Network model performance was measured using the overall average platform counts at all Toronto subway platforms (Fig. 8). The figure shows a model network that is able to relatively quickly recover (within 10 min) to normal crowding levels on the platform for short-duration disruptions (10 min) without any change to train operation. Disruptions of longer duration (20 and 30 min) were not as well handled without intervention by the transit service. A disruption of 20 min (7:55–8:15 AM) required 25 min after the end of disruption to return to normal levels of platform use. At a disruption of 30 min, however, the network was unable to recover, maintaining elevated passenger crowding levels on the platforms throughout the remainder of the simulation period. For both of these longer disruption lengths, as expected, the model produced agent travel times that were noticeably longer, particularly for the 30 min disruption.

Overall, the case study model acted as expected when stressed with disruption events of varying lengths. These situations were also able to be analyzed without

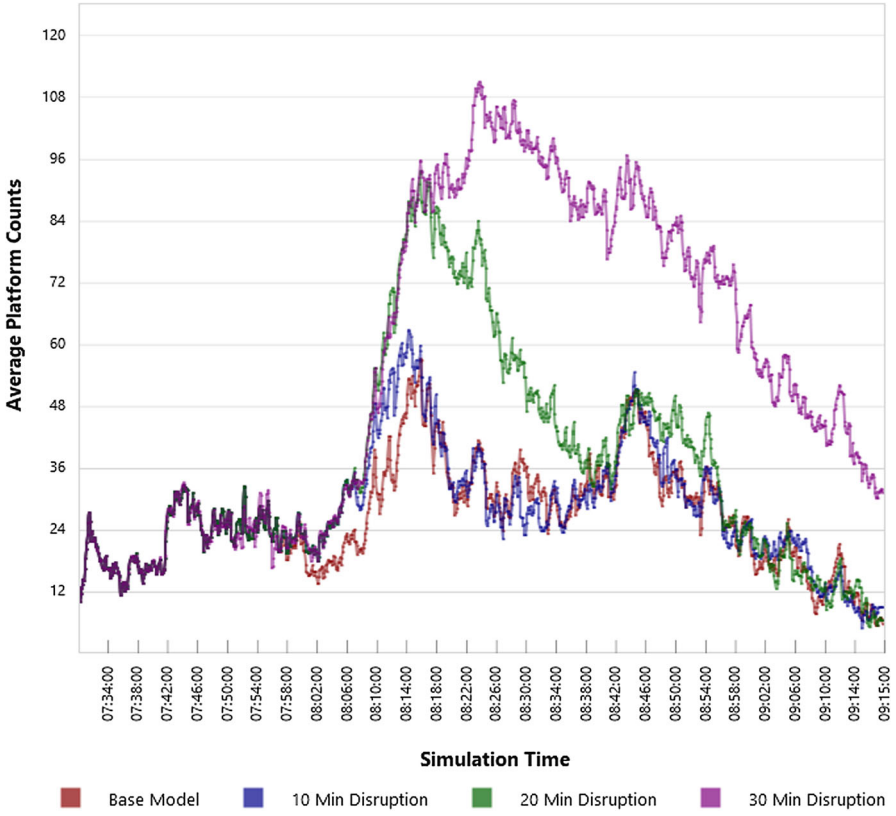

Fig. 8 Impact on average platform counts system-wide from disruptions of various durations at Bloor/Yonge station

requiring modifications to the base model files with on-the-fly commands given to the line simulator. Finally, the results show that the platform has its use in both understanding the impact of specific disruptions, but also in determining the level of stress the network could handle before intervention is needed.

11 Case study: testing a disruption management strategy

Finally, to show proof-of-concept of how Nexus could be used to conduct on-the-fly response strategy testing, a simple response measure was devised for the longest disruption scenario analyzed in the prior section (30 min hold of a train at the southbound platform of Bloor/Yonge station). To maintain simplicity, train operation was left as is; modification to train schedules (e.g. turning them back) would require creating new trip schedules for each vehicle both during and after the disruption, a non-trivial task. Instead, the response measure analyzed focussed on provision of information to relevant agents. The goal of the response measure was to avoid the over-crowding situation at the southbound platform due to a continuous influx of agents from the westbound Bloor-Danforth line. Agents who would be projected to make the transfer to the southbound platform at Bloor/Yonge station were made aware of the disruption and allowed to reconsider their route choice. This decision point was set as the time of arrival of their current vehicle at its next stop.

Figures 9 and 10 show the impact of the response on a relevant performance measure. Average platform crowding (Fig. 9) displayed a noticeable reduction with the response method. However, the majority of this was attributable to the greatly reduced passenger flow through the southbound platform at Bloor/Yonge (Fig. 10). As the agents affected were predominantly only those transferring from the westbound direction of the Bloor-Danforth line, and no adjustments to train or bus service was made, the response strategy's minimal impact on other platforms was expected. The response strategy, while reducing overall platform crowding, could not return overall levels to normal during the simulation period.

This final investigation illustrated the capability of Nexus in assisting in response strategy testing for disruptions. It also displayed an ability of the system to incorporate or be compatible with a dynamic transit assignment method, which has relevance beyond disruption management scenarios to evaluating more general transit ITS information provision technologies.

12 Notes on validating a large-scale microscopic model

For any simulation model, proper calibration of parameters and validation of simulation results is essential in order to use the model for predictive analysis. However, as a proof-of-concept example of an early-stage prototype system, particularly with a simplified transit assignment module, this was not a focus of the case study presented above. Several steps are required before full validation can occur. This will include replacing the all-or-nothing transit assignment method with

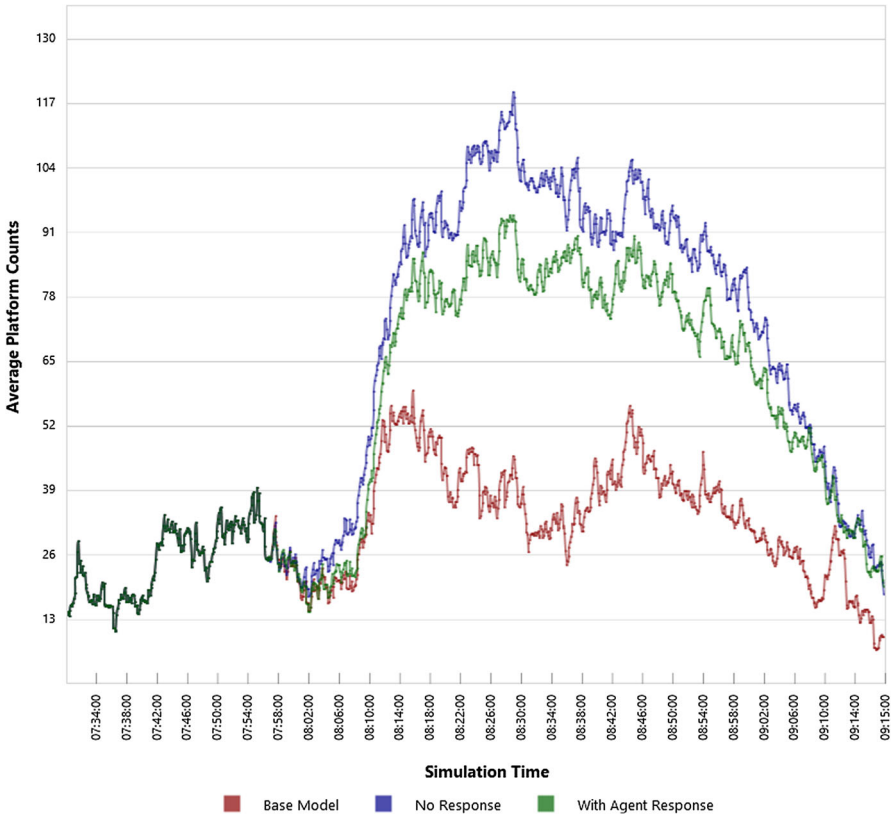

Fig. 9 Impact of response strategy on average platform counts across the subway network

MILATRAS, and replacing the stand-in models for surface, line and station by commercial software or more detailed models, including MassMotion models of other key stations. While such tasks are expected to occur over the long-term, this does not mean that validation has to be put off until all pieces of the model are fully fleshed out.

There are two categories of validation that can be performed for a model in Nexus, validation of the entire network including all movements of transit users and service, and validation of the individual components of the network, including specific behavioural models. Validation of individual components (e.g. vehicle movement in between stations, pedestrian flow inside stations) or of specific behavioural processes (e.g. local or network route choice by agents) can be done independently. The ability of Nexus to allow for separate models to be linked allow for this validation to be done separately within the component software. For example, in the case study system, a detailed MassMotion model of Yonge/Bloor station, calibrated and validated in a prior study, was brought in to be linked in with the rest of the network model, maintaining parameter values that were shown to produce realistic routing of agents.

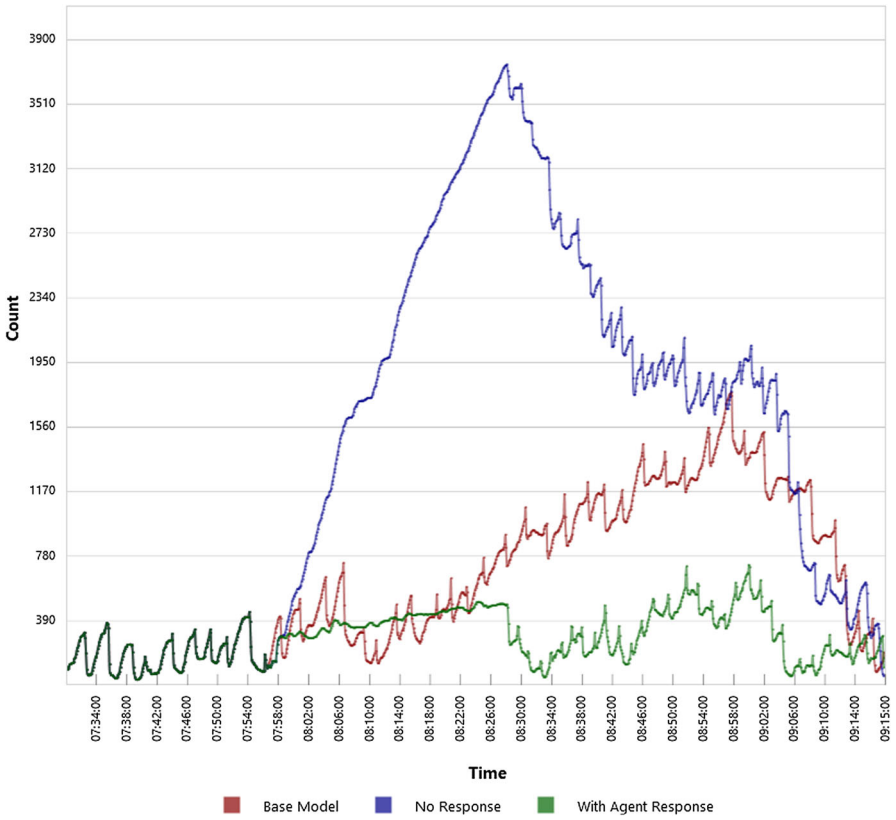

Fig. 10 Impact of response strategy on platform counts at Bloor/Yonge southbound

Validation of the overall network, however, is more challenging and dependent on all pieces being modelled appropriately. It is possible, once a load-balancing transit assignment method is incorporated, to validate route loads against publicly available data; this is expected to be the level of validation most feasible in the near term as work continues on developing the model. Microscopic validation at a network-scale is more difficult, but is feasible. This would involve validation of models of interaction at component interfaces (e.g. dwell processes at train platforms) against field data, and acquiring micro-level data at a network-scale from agencies for a more detailed examination. Such data would include train location data, AVL for surface vehicles, APC and smart card data for vehicle loads and to/from stations, and in-station counts at other key locations (e.g. stairs and escalators). They should be obtained for both regular and disrupted operations in order to validate the model's ability to predict the impact of disruptions and service response.

13 Conclusion

This paper presents a proof-of-concept case study of the Toronto transit network to illustrate the capabilities of the Nexus transit simulation platform prototype. The main goals were to show its ability to model a large-scale network, interface with a commercial pedestrian simulator, and be used to simulate disruptions and test response strategies. Using data from a variety of sources, including GTFS and data from a recent regional travel survey, a model was constructed of the Greater Toronto transit network during the AM peak period. This included full 3D models, simulated within MassMotion, of two key transfer stations. Using this base model, an analysis of the network-level impact of disruptions on transit service and platform crowding was conducted, as well as illustrating how Nexus could be used to test a possible disruption response strategy. While a greatly simplified model in many aspects, this exercise produced some tentative takeaways, namely that even without any intervention, the subway system could absorb passengers built up during a disruption of reasonable length, and a low cost solution (simple announcement asking passengers to find an alternate route) could help alleviate dangerous crowding levels.

The larger implications for disruption managers are not, however, in the specific outcomes of the case study, but in the eventual capabilities of Nexus in aiding disruption management. As an extension of the stress test presented in the case study, Nexus will enable determining critical points of failure within a train network, those that would cause the most significant disruption, at a level of detail currently unavailable. Also possible are evaluations of strategies to combat the resulting build-up in crowding, for example through offering or making passengers aware of alternative paths, better positioning of possible response assets, or crowd control measures. This, at first, will most likely be possible in a training environment, allowing tests of responses in a simulated setting in place of the ad hoc approach today. However, significant research efforts remain, particularly in more accurately modelling detailed rail operation, incorporating a more sophisticated transit assignment procedure, and implementing capabilities to model responses utilizing shuttles and vehicle rescheduling, to fully realize the disruption management capabilities of the platform.

Acknowledgements The authors wish to acknowledge the support provided by Arup. The work would also not have been possible without Canadian federal funding provided by the Natural Sciences and Engineering Research Council (NSERC).

References

Angeloudis P, Fisk D (2006) Large Subway Systems as Complex Networks. Physica A: Statistical Mechanics and its Applications 367:553–558

Cats O (2011) Dynamic Modelling of Transit Operations and Passenger decisions. PhD thesis, KTH Royal Institute of Technology

Chen A, Yang C, Kongsomsaksakul S, Lee M (2007) Network-based Accessibility Measures for Vulnerability Analysis of Degradable Transportation Networks. Networks and Spatial Economics 7(3):241–256

Cortes CE, Burgos V, Fernandez R (2010) Modelling Passengers, Buses and Stops in Traffic Microsimulation: Review and Extensions. Journal of Advanced Transportation 44(2):72–88
Harris N, Anderson R (2007) An International Comparison of Urban Rail Boarding and Alighting Rates. Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit 221(4):521–526
Jespersen-Groth J, Potthoff D, Clausen J, Huisman D, Kroon L, Maróti G, Nielsen MN (2009) Disruption Management in Passenger Railway Transportation. Robust and Online Large-Scale Optimization. Lect Notes Comput Sci 5868:399–421
Kelin U, Schulze T, Straßburger S (1998) Traffic Simulation Based on the High Level Architecture. Proceedings of the 1998 Winter Simulation Conference 2:1095–1103
King D, Srikukenthiran S, Shalaby A (2014) Using Simulation to Analyze Crowd Congestion and Mitigation at Canadian Subway Interchanges. Transportation Research Record: Journal of the Transportation Research Board 2417(Transit 2014, Vol 3):27–36
Klefstad R, Zhang Y, Lai M, Jayakrishnan R, Lavanya R (2005) A Distributed, Scalable, and Synchronized Framework for Large-Scale Microscopic Traffic Simulation. In: 8th International IEEE Conference on Intelligent Transportation Systems, IEEE, pp 813–818
Lämmel G, Chraibi M, Wagoum AUK, Steffen B (2016a) Hybrid Multimodal and Intermodal Transport Simulation. Transportation Research Record: Journal of the Transportation Research Board 2561:1–8
Lämmel G, Klüpfel H, Nagel K (2016b) The MATSim Network Flow Model for Traffic Simulation Adapted to Large-Scale Emergency Egress and an Application to the Evacuation of the Indonesian City of Padang in Case of a Tsunami Warning. In: Pedestrian Behavior, Emerald Group Publishing Limited, pp 245–265
Lee K, Jung WS, Park JS, Choi MY (2008) Statistical Analysis of the Metropolitan Seoul Subway System: Network Structure and Passenger Flows. Physica A: Statistical Mechanics and its Applications 387(2):6231–6234
Liu H, Ma W (2005) Jayakrishnan R (2005) Distributed Large-Scale Network Modeling with Paramics Implementation. IEEE Intelligent Transportation Systems Proceedings 2005:232–238
Morrow E (2011) Efficiently Using Micro-Simulation to Inform Facility Design – A Case Study in Managing Complexity. Pedestrian and Evacuation Dynamics. Springer, Boston, MA, pp 855–863
Patel A, Hudson-Smith A (2011) Agent Tools, Techniques and Methods for Macro and Microscopic Simulation. Agent-Based Models of Geographical Systems. Springer, Dordrecht, pp 379–407
PTV (2012) VISSIM 5.40 User Manual. epubli
Rieser M (2010) Adding Transit to an Agent-Based Transportation Simulation. PhD thesis, Swiss Federal Institute of Technology
Sienkiewicz J, Holyst JA (2005) Statistical Analysis of 22 Public Transport Networks in Poland. Phys Rev E 72(4):046,127
Srikukenthiran S, Shalaby A, (2011) Scalable Microsimulation Modelling Framework of Crowd and Subway Network Dynamics for Improved Disruption Management. In: Computers in Urban Planning and Urban Management, (2011) Conference. Lake Louise, Alberta
Srikukenthiran S, Shalaby A (2012) Prototyping a Scalable Agent-based Modelling Framework for Large-Scale Simulation of Crowd & Subway Network Dynamics. In: Conference on Advanced Systems in Public Transport 2012, Santiago, Chile
Srikukenthiran S, Shalaby A (2015) Time-based Model of passenger Dispersion on Subway Platforms. In: Transportation Research Board 94th Annual Meeting, Washington, DC
Srikukenthiran S, Shalaby A, Morrow E (2014) Mixed Logit Model of Vertical Transport Choice in Toronto Subway Stations and Application within Pedestrian Simulation. Transportation Research Procedia 2:624–629
Still GK (2000) Crowd Dynamics. PhD thesis, University of Warwick
Su F, Roorda MJ, Miller EJ, Morrow E (2015) An Integrated Approach to Estimate Pedestrian Exposure to Vehicle Emissions. In: Transportation Research Board 94th Annual Meeting
Toronto Transit Commission (2008) Toronto Transit Commission Contract S85–40. Tech. rep, Toronto
von Ferber C, Holovatch T, Holovatch Y, Palchykov V (2009) Modeling Metropolis Public Transport. Traffic and Granular Flow '07 (Chapter 80):709–719
Wahba M, Shalaby A (2005) Multiagent Learning-Based Approach to Transit Assignment Problem: A Prototype. Transportation Research Record: Journal of the Transportation Research Board 1926:96–105

Wahba M, Shalaby A (2011) Large-scale Application of MILATRAS: Case Study of the Toronto Transit Network. Transportation 38(6):889–908

Wang J, Wahba M, Miller EJ (2010) Comparison of Agent-Based Transit Assignment Procedure with Conventional Approaches. Transportation Research Record: Journal of the Transportation Research Board 2175:47–56