Journal of Transport Geography 62 (2017) 92–97

Contents lists available at ScienceDirect

Journal of Transport Geography

journal homepage: www.elsevier.com/locate/jtrangeo

Constructing a routable retrospective transit timetable from a real-time vehicle location feed and GTFS

Nate Wessel*, Jeff Allen, Steven Farber

University of Toronto, Sidney Smith Hall, 100 St. George Street, Room 5047, Toronto M5S 3G3, Ontario, Canada

A B S T R A C T

We describe a method for retroactively improving the accuracy of a General Transit Feed Specification (GTFS) package by using a real-time vehicle location data set provided by the transit agency. Once modified, the GTFS package contains the observed rather than the scheduled transit operations and can be used in research assessing network performance, reliability and accessibility. We offer a case study using data from the Toronto Transit Commission and find that substantial aggregate accessibility differences exist between scheduled and observed services. This 'error' in the scheduled GTFS data may have implications for many types of measurements commonly derived from GTFS data.

1. Introduction

Over the last ten years, the General Transit Feed Specification (GTFS) has emerged as an industry standard for publishing data about transit operations. Data in this format has issued from more than a thousand transit agencies around the world and that data has been incorporated into just as many user-facing routing applications. GTFS data defines transit schedule information in a format that is essentially a routable spatiotemporal network graph with stops as nodes, scheduled travel between stops as edges, and estimated travel times as the cost. This not only allows people to find their way from A to B, but due to the open nature of the standard, has allowed researchers to ask interesting questions and have them answered with a degree of accuracy and scope that would have been impossible before GTFS. Such questions, still very much under active research, include measures of disparities in service provision (Farber et al., 2016; Fransen et al., 2015), temporal variability (Farber et al., 2014), the role of relative travel times and costs in mode choice (Owen and Levinson, 2015; Salonen and Toivonen, 2013), the degree of accessibility offered by competing transit development plans (Farber and Grandez, 2017), and many others. Yet, such research using GTFS is subject to a serious criticism: it is based entirely on schedules, which are expectations about services, rather than observations of them. It is common knowledge that transit does not run precisely as scheduled, and that it often differs substantially from the schedule. Occasionally there are major unscheduled disruptions due to severe congestion, vehicle breakdowns, or signal malfunctions. These disruptions are a fact of life for most transit users and well acknowledged by transit agencies themselves.

One way that agencies have acknowledged service delays and disruptions is to issue live updates to their transit schedules. While for some smaller agencies, these take the form of hastily posted signs and twitter notifications, for many larger operations, the effort to update their customers has become perpetual. We refer here to the "real-time" data sources that contain constantly updated arrival predictions as well as live vehicle location reports based on Geographic Positioning Systems (GPS). These algorithmically produced updates have proven quite useful to many transit users (Brakewood et al., 2015; Tang and Thakuriah, 2012; Watkins et al., 2011), and the automatic vehicle location (AVL) systems that enable them have proven useful to transit researchers who have primarily used them to assess reliability at the level of stops or lines (e.g. El-Geneidy et al., 2011; Tribone et al., 2016) or to propose ways for transit operators to improve reliability or prediction accuracy. Yet transportation researchers have not yet to our knowledge made use of such data to answer research questions that involve routing across the transit network, a type of query enabled by GTFS data.

In this paper, we propose a novel way of making this "real-time" data available to answer precisely the same sorts of research questions that people have already been asking of and answering with GTFS. We do this by using it to update an existing, schedule-based GTFS package with observed trips and arrival times, yielding a GTFS package based on observation rather than schedule. Of course, such data cannot be used directly for routing actual passengers since the events it describes will already have transpired. Yet, most research questions currently answered by GTFS data might be better directed not at a schedule but at a measure of average performance which could be derived from past

* Corresponding author.
E-mail address: nate.wessel@mail.utoronto.ca (N. Wessel).

http://dx.doi.org/10.1016/j.jtrangeo.2017.04.012
Received 18 January 2017; Received in revised form 26 April 2017; Accepted 26 April 2017
Available online 09 June 2017

events, or at the events of some particular day or time in the past.

In the remainder of this paper, we will describe our method for creating this retrospective GTFS package from GTFS and real-time data. We will then undertake a brief case study of the Toronto Transit Commission to illustrate the potential utility of this approach. Because real-time data is currently not well standardized, we will have to describe our algorithm to some extent as it was designed around a particular standard - the data available from the NextBus company's API (NextBus, 2016) which is used in Toronto and many other cities. The code we developed is available online,[1] and we encourage others to use and contribute to the project. Work to extend the code to accommodate other real-time data standards is underway, and we will try to describe the project here with a degree of abstraction sufficient to allow the reader to see how the technique can be applied to any real-time data standard.

2. Required data

Our method relies upon three data sets:

1. A current GTFS package, which will be used primarily to provide the locations of stops and stations.
2. A real-time feed of vehicle locations which are preferably associated with some route information.
3. A routable road and/or rail network data set including all ways used by transit vehicles.

The method we have developed is based on the information provided by a basic GTFS package, and the real-time information provided by the API delivered by the NextBus company, which currently serves data for about 50 agencies around the world (NextBus, 2016). Other real-time data formats exist, and real-time data has not yet standardized to the same degree as schedule data; some agencies provide different information in their feeds. We believe that our algorithm should be generalizable to most real-time data feeds, so long as they provide accurate vehicle locations and update them at a sufficient frequency.

3. Algorithm

3.1. Outline

The basic ontological units of a GTFS package are routes, trips, blocks, and stops. Routes are sets of typical service patterns grouped under a single name, usually a number. Trips are particular occasions when a vehicle goes from one end of a route to the other, serving an ordered set of stops. Blocks are sets of trips in the order of their performance, operated continuously by the same vehicle.[2] Stops are the locations where passengers can access a trip.

Real-time data often does not describe the transit system in the same terms as does GTFS; it is not designed to provide a routable network, but to give updates on particular vehicles and lines. The task of our algorithm then must be to translate the description provided by the real-time data into the terms/units used by GTFS. In the real-time data provided by NextBus for example, there is no explicit concept of trips or blocks and these must be inferred. The focus of the NextBus data is on vehicles and stops.

Our basic approach is to monitor vehicles in real-time as their locations are updated, keeping track of where they are and when. When we observe that a vehicle seems to have completed a trip, we process the data associated with that trip and begin building up a new trip for the vehicle if it is still in service. Finished trips are associated with stops from the schedule data and stop times are derived from the timestamps of vehicles passing those stops. Next, trips are assigned to blocks and routes and the data is stored in a database. At this point, it is straightforward to extract the data in the text-based GTFS format. Much of the hard work of the algorithm is in error handling and data cleaning as reported vehicle locations are always a bit fuzzy and occasionally spurious.

3.2. Collecting and storing the data

We developed a program in Python to collect real-time data from the NextBus API, which is used by the Toronto Transit Commission. The NextBus API is a publicly available web service designed to serve real-time transit data primarily to mobile phone or web applications. One of its functions is to report the latest locations for all operating vehicles in the fleet, the idea being that these will be displayed to users on a mobile web map. For each vehicle, the API returns information on the vehicle's ID, route, heading, last known location, and the time it reported that location to the server. Vehicles update their location every 20 s on average.

Our program requests all updated vehicle locations every 10 s to ensure that all new data is collected and then stores the locations in a PostGIS database along with a timestamp and the other associated information.

3.3. Delimiting trips and blocks

We assume that each vehicle reported to be in service is operating a trip and that that trip belongs to a schedule block. Each vehicle under observation must be assigned a unique trip id and a unique block id. A block ends only when the vehicle goes out of service, defined as failing to report a location for three or more minutes (or some other threshold, as appropriate). The NextBus API did not explicitly report when a vehicle was going out of service. The end of a schedule block implies the end of any ongoing trip, but a trip may also end in a number of other conditions. For the data available to us from NextBus, we defined these as

1. The vehicle reporting a different route.
2. The vehicle reporting a different headsign.

Headsigns typically indicate the direction of travel, i.e. “501 E” and a change to “501 W” implies that the terminal station has been reached and the return trip begun in the opposite direction. With the NextBus data, we were fortunate that the headsign and route changes were a good indicator of trip endings. Other real-time or historical data sets could necessitate other techniques, perhaps simpler, perhaps not, for determining when trips are ending. For pure GPS data, it may be necessary to determine when a vehicle turns back on its route to go the other way.

Once a trip has ended, all information associated with that trip is pulled from the database and processed, as the rest of this section will describe. If the vehicle is still in service, a new trip is started for that vehicle, and new location reports begin to accumulate in the database associated with the new trip ID.

3.4. Spatial matching and positional error handling

Visual inspection of the data provided by the NextBus API showed normal, expected levels of GPS position error for most points, but also some extreme errors associated with trips either starting or ending in a bus garage, where vehicles reported positions many kilometres from the last reported location before going out of service. The extreme errors were easily dealt with by measuring the speed (*distance/time*) between location reports. Segments over 120 km/h indicated obvious errors and the offending points were removed accordingly. As these were almost always associated with vehicles entering or leaving service, this was not

[1] https://github.com/SAUSy-Lab.

[2] Passengers can often stay on the vehicle as it transitions between trips.

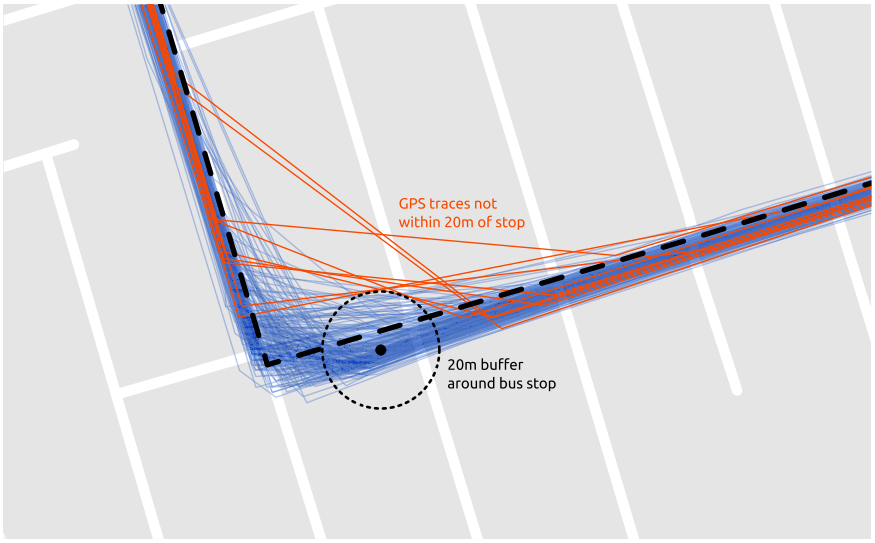

Fig. 1. Map showing GPS tracks overlaid on street centerlines. Positional error and low sampling rates mean the GPS tracks may be poorly related to the actual path taken by the vehicle.

a major problem.

Ordinary GPS measurement error did present a more serious issue when coupled with the relatively low sampling rate of the data (about every 20 s). There were often times when no location would be reported near a stop as the vehicle must have passed that stop quickly. This was particularly problematic where the stop was positioned just before or after a turn in the route, as can be seen in Fig. 1.

To improve the spatial resolution of the data throughout the duration of the observed trip, and to catch any substantial positional errors that remained, we matched the sampled locations of the trip to a plausible route on the network. Map-matching is a common task in modern on-board navigation systems and the problem has been well researched (e.g. Krumm et al., 2007; Lou et al., 2009; Newson and Krumm, 2009).

We matched our trips to a combination of roads and streetcar tracks with detailed data from OpenStreetMap[3] using the Open Source Routing Machine (OSRM)[4], an open-source routing software. An example of the matched result compared to an original GPS input can be seen in Fig. 2.

OSRM accepts a series of points and (optionally) timestamps and positional error estimates which it uses to match the route to the provided network. It also allows for custom vehicle profiles; we developed one for transit vehicles which gave us the ability to match to streetcar tracks and agency-managed service roads, as well as to adjust expectations about realistic expected travel speeds. For each returned match, OSRM also estimates the probability that the returned route is the correct one based on a number of factors using a naive Bayes classifier. Visual inspection of many thousands of matched geometries indicated that these estimates were very effective at picking out erroneous matchings produced by positional or network data errors.

At first, as many as 10% of our trips were able to find only very poor matches ($P < 0.5$). Visual inspection revealed that most of these were attempting to pass through intersections tagged with incorrect turn or access restrictions. Once these errors were corrected in the OpenStreetMap data, our error rate (again, $P < 0.5$) fell to around 1.5% of all trips. These badly mismatched trips appear to be mostly the result of GPS noise in urban canyons, and for the purpose of this analysis such trips were discarded. Further refinement of either the routing profile or the street network data set could almost certainly increase the match rate further, though we found the 1.5% error rate acceptable for the purpose of demonstrating the technique here and do not see reason to suspect that it substantially influenced the case study which follows. Removed trips were not concentrated in any one part of the city to a great degree.

3.5. Determining stop times

Once we have a trip matched to the network, with timestamps associated with matched vehicle locations on that route, the next step is to determine which stops were passed and estimate their times of arrival and departure. From the trips' reported heading and route ID, we can find a plausible set of stops from the schedule-based GTFS package. This is especially easy if the metadata associated with vehicles in the real-time data provides trip ids that match those in the schedule-based GTFS, as would be the case for the GTFS-Realtime standard developed by Google. In our case, the set of stops being serviced came from the NextBus API's routeConfig command,[5] though these were identical to those provided in the GTFS package. From these stops, only those within 20 m of the matched route of the vehicle are used and these are then matched to the nearest point on that route. This is an important step, as many vehicles may not operate their full route, either by taking a detour around a blockage or by making a short-turn to reduce bunching.

The time for the stop can then be estimated by linear interpolation from the surrounding vehicle reports (Fig. 3). Ideally, it should be possible also to estimate dwell times if there are two or more location reports in the vicinity of the stop. These would then be encoded as different arrival and departure times for the stop. In practice, we found this difficult to implement, and as long as the departure time is more or less correct, the dwell time would make little difference to any routing on the network.

3.6. Constructing the retrospective GTFS package

GTFS is a text-based format with a specified table structure[6] and comma-delimited files. Up to this point, we have not specified how data should be stored as it is processed. In our implementation, all data is stored in a PostGIS database, where some of the spatial operations are executed. This database is structured in essentially the same way as GTFS data with a table each for trips, stop times, routes, stops, and service dates. Since our data is essentially already in a tabular format, we need only select all required fields and format them according to the

[3] http://www.openstreetmap.org.
[4] http://project-osrm.org/.
[5] https://www.nextbus.com/xmlFeedDocs/NextBusXMLFeed.pdf.
[6] https://developers.google.com/transit/gtfs/reference/.

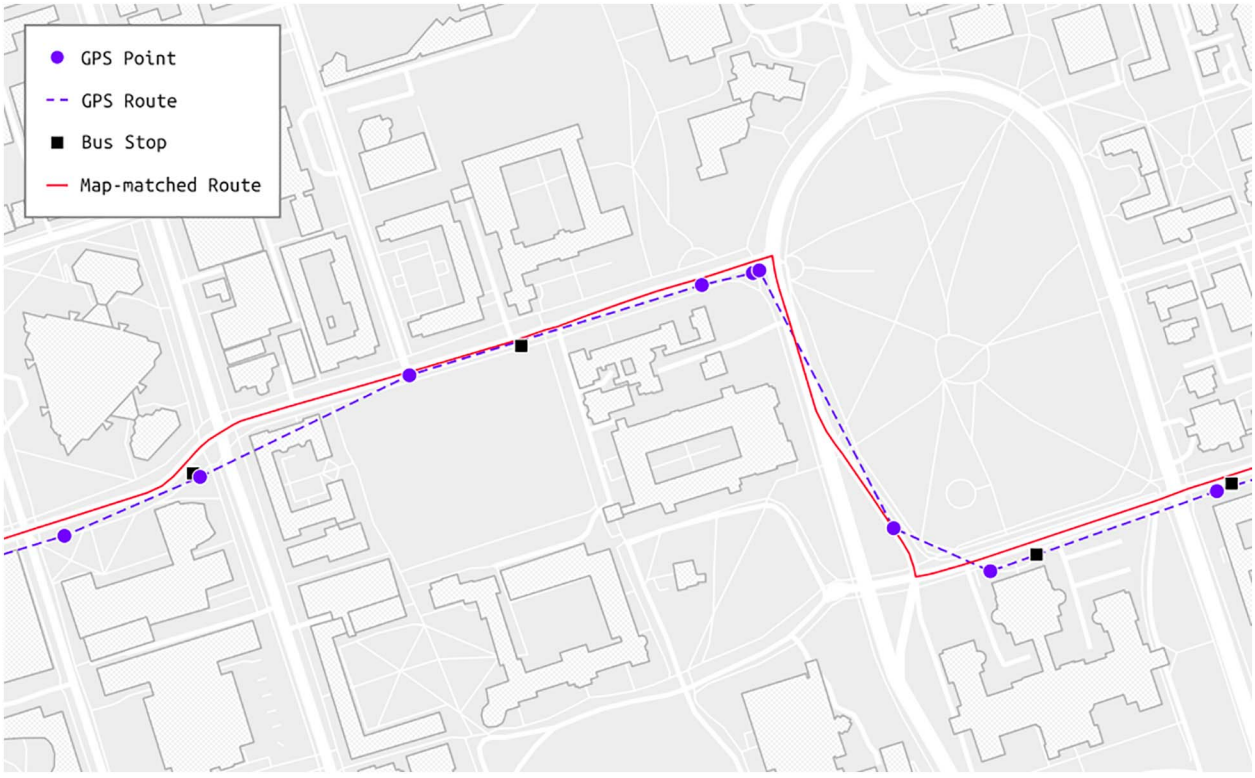

Fig. 2. Map showing network path (red, solid) derived from example GPS coordinates (purple, dashed). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

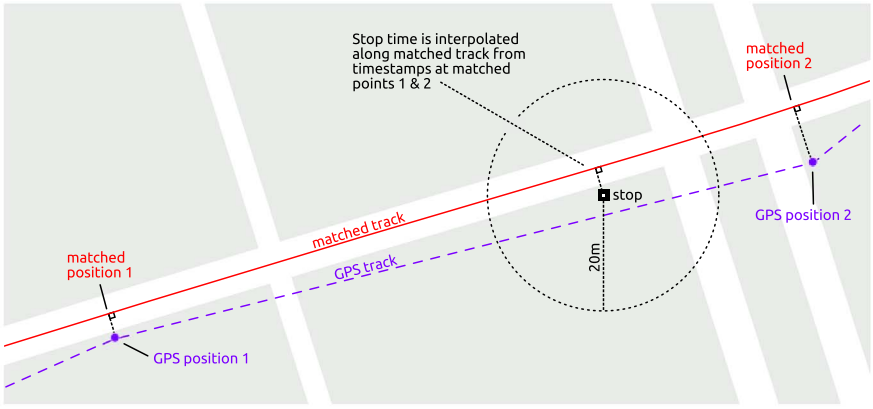

Fig. 3. Diagram showing interpolation method

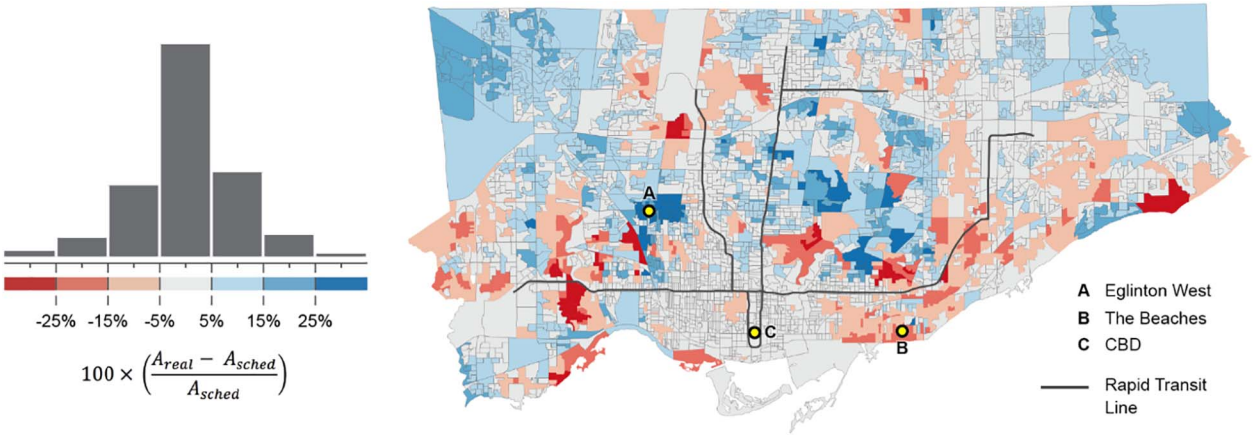

Fig. 4. Relative (percent change) differences between scheduled and real-time access scores. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

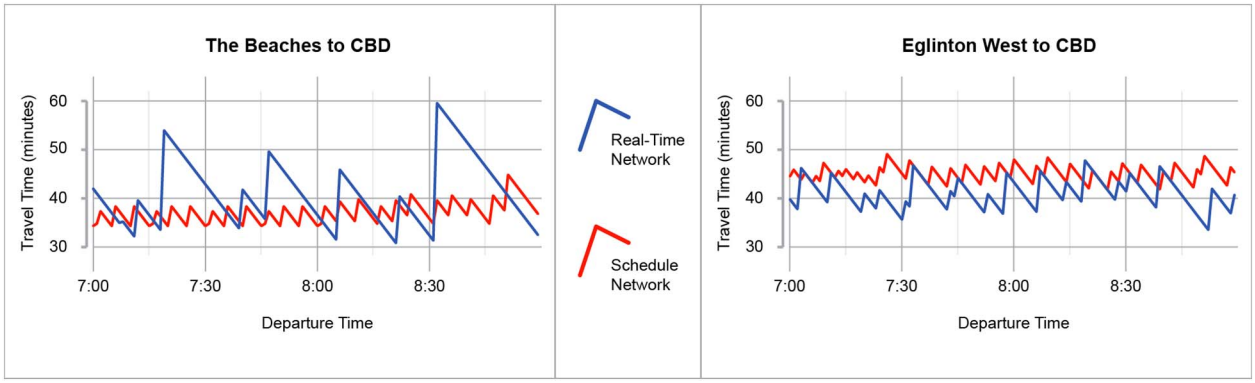

Fig. 5. Scheduled and real-time travel times resulting in accessibility differences.

GTFS documentation. A distinct service ID is generated for each day of observed data. All trips with more than two observed stops are included.

4. Accessibility case study

We now demonstrate the potential utility of the retrospective GTFS by comparing a simple measure of average accessibility to jobs in Toronto between scheduled and retrospective data sets for the same period. The Toronto Transit Commission (TTC) operates four rapid transit lines as well as a grid network of high frequency surface bus and streetcar routes. Employment in Toronto is concentrated in the Central Business District (CBD) and there is high residential density near the CBD, with lower density residential neighbourhoods in the rest of the city. Public transit mode share in Toronto for commuting trips is 36.55% (Statistics Canada, 2011).

Accessibility to jobs via transit is a measure that has recently been computed using scheduled GTFS packages (Farber and Fu, 2017; El-Geneidy et al., 2016; Owen and Levinson, 2015). For our case study, we computed travel times from all census dissemination areas (DAs) in Toronto to all traffic analysis zones (TAZs). This was done using OpenTripPlanner[7], an open source multimodal routing engine, which computed centroid-to-centroid travel times inclusive of walking, waiting, in-vehicle, and transferring times. From each DA we calculated the total number of jobs (aggregated to TAZs) accessible within 45 min of travel time, for every minute in the morning rush-hour period (7 am–9 am), over two days of both observed and scheduled data. This resulted in a set of time-varying accessibility scores for each DA, which we then averaged. The difference between the average accessibility between the retrospective and scheduled data sets is shown in Fig. 4. Unfortunately, real-time data were not available for the TTC's four rapid transit lines (black lines in Fig. 4), so for these we simply copied the scheduled GTFS data for these routes in our real-time GTFS package prior to our analysis.

Negative values in Fig. 4 (red) denote locations where accessibility scores using the real-time network are lower than those using the scheduled network, and indicate where observed service culminated in lower than scheduled levels of accessibility to jobs. Positive values (blue) denote the opposite; these are areas where observed accessibility levels are higher than those obtained by using the scheduled network.

It may perhaps come as a surprise to some readers that actual operations could result in greater average local accessibility than strict adherence to the schedule. This is likely due to conservative schedules, expecting delays that were not actually encountered (Wessel and Widener, 2016). It is also worth noting that in Toronto's case, many lines are not actually operated with the goal of adhering to a published schedule; rather the operational goal is to maintain adequate headways between vehicles, especially during peak hours. In this case, the speed of vehicles across the route is not explicitly defined and drivers will likely go as fast as traffic permits while still maintaining space from the leading vehicle. Such operational practices (headway maintenance versus schedule adherence) can be indicated in GTFS schedules,[8] yet many agencies do not make use of these optional features.

The two accessibility scores obtain similar values nearer to the city centre and along the rapid transit lines, largely because the information on these lines is identical between the two packages. Differences in accessibility are clustered in certain neighbourhoods and along specific routes where transit operations appear to differ substantially from the published schedule. This is further visualized in Fig. 5 with two comparative minute-by-minute travel time plots, each from a residential neighbourhood (The Beaches and Eglinton West) to Toronto's Central Business District (CBD).

The left plot is an example of where transit operates with greater headways and results in longer trips compared to the published schedule. Presumably there were less transit vehicles in operation than scheduled during this period or there was severe vehicle bunching. Conversely, the plot on the right is an example of where travel times from the real-time network are on average lower than those from the scheduled network. This could be because the agency overestimated operational delays when scheduling their service.

Interestingly, our case study finds that the aggregate population-weighted accessibility scores for the entire city are almost equal when comparing the two networks. For the 45 minute travel time threshold shown in Fig. 4, the scheduled data set had an overall average accessibility score of 268,100 jobs, while the real-time data set resulted in a score of 269,022 jobs; a difference of only 0.34%. Similar results were found when testing different travel time thresholds.

5. Discussion

What is interesting in this case study is not that there is substantial deviation from the schedule, but that this deviation does not seem to be random. Our measure of cumulative accessibility, even when averaged over four hours of peak period service from two days of operation showed what appears to be systematic variation that affects neighbourhoods differently. This means that measures of accessibility, travel time, etc. derived from GTFS data may not be a good estimate of actual values even when averaged.

It would not be fair to characterize such non-adherence to a GTFS schedule as an operational failure. Transit systems were operating before GTFS data came into being and many of them are still operating in the same way. For example, scheduled transit is often operated by timepoints, usually a small subset of stops at which strict schedule adherence is desirable. Arrival times for stops between these are

[7] http://www.opentripplanner.org/.

[8] https://developers.google.com/transit/gtfs/reference/frequencies-file.

undefined in practice yet GTFS demands that a precise arrival and departure time be given for these stops. In reality, drivers may make no attempt to adhere to these times and may not even be aware of them. Similarly, we should be careful before interpreting deviations from GTFS schedules with a spatial equity lens. GTFS data is provided by agencies primarily to enable customer-facing trip-planning applications, not to enable researchers to conduct various kinds of accessibility analysis with a high degree of accuracy. The goal of our work is to provide a data set more suitable for such research applications.

As researchers continue to make use of the wealth of data available via GTFS, they should be aware that such data may differ systematically from actual transit operations. Measures derived from scheduled GTFS data alone may be subject to bias if there is a mismatch between the data and the operational reality.

References

Brakewood, C., Macfarlane, G.S., Watkins, K., 2015. The impact of real-time information on bus ridership in New York City. Transp. Res. C Emerg. Technol. 53, 59–75.

El-Geneidy, A., Buliung, R., Diab, E., van Lierop, D., Langlois, M., Legrain, A., 2016. Non-stop equity: assessing daily intersections between transit accessibility and social disparity across the Greater Toronto and Hamilton Area (GTHA). Environ. Plann. B. Plann. Des. 43.3, 540–560.

El-Geneidy, A.M., Horning, J., Krizek, K.J., 2011. Analyzing transit service reliability using detailed data from automatic vehicular locator systems. J. Adv. Transp. 45.1, 66–79.

Farber, S., Fu, L., 2017. Dynamic public transit accessibility using travel time cubes: comparing the effects of infrastructure (dis)investments over time. Comput. Environ. Urban. Syst. 62, 30–40.

Farber, S., Grandez, M., 2017. Transit accessibility, land development and socioeconomic priority: a typology of planned station catchment areas in the Greater Toronto and Hamilton Area. J. Transp. Land Use 10 (1), 33–56.

Farber, S., Morang, M.Z., Widener, M., 2014. Temporal variability in transit-based accessibility to supermarkets. Appl. Geogr. 53, 149–159.

Farber, S., Ritter, B., Fu, L., 2016. Space-time mismatch between transit service and observed travel patterns in the Wasatch Front, Utah: a social equity perspective. Travel Behav. Soc. 4, 40–48.

Fransen, K., Neutens, T., Farber, S., De Maeyer, P., Deruyter, G., Witlox, F., 2015. Identifying public transport gaps using time-dependent accessibility levels. J. Transp. Geogr. 48, 176–187.

Krumm, J., Horvitz, E., Letchner, J., 2007. Map Matching with Travel Time Constraints. Tech. rep. SAE Technical Paper.

Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y., 2009. Map-matching for low-sampling-rate GPS trajectories. In: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, pp. 352–361.

Newson, P., Krumm, J., 2009. Hidden Markov map matching through noise and sparseness. In: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, pp. 336–343.

NextBus, 2016. Public xml Feed Documentation. https://www.nextbus.com/xmlFeedDocs/NextBusXMLFeed.pdf.

Owen, A., Levinson, D.M., 2015. Modeling the commute mode share of transit using continuous accessibility to jobs. Transp. Res. A Policy Pract. 74, 110–122.

Salonen, M., Toivonen, T., 2013. Modelling travel time in urban networks: comparable measures for private car and public transport. J. Transp. Geogr. 31, 143–153.

Statistics Canada, 2011. 2011 National Household Survey.

Tang, L., Thakuriah, P.V., 2012. Ridership effects of real-time bus information system: a case study in the city of Chicago. Transp. Res. C Emerg. Technol. 22, 146–161.

Tribone, D., Authority, M.B.T., Riegel, L., Warade, R., Barker, D., 2016. Measuring transit agency performance using open realtime data. In: Transportation Research Board 95th Annual Meeting. 16-5080.

Watkins, K.E., Ferris, B., Borning, A., Rutherford, G.S., Layton, D., 2011. Where is my bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. Transp. Res. A Policy Pract. 45.8, 839–848.

Wessel, N., Widener, M., 2016. Discovering the space-time dimensions of schedule padding and delay from GTFS and real-time transit data. J. Geogr. Syst. 1–15. http://dx.doi.org/10.1007/s10109-016-0244-8. ISSN 1435-5949.