FastTrack: An Optimized Transit Tracking System for the MBTA

A Group of 3 MIT Students in 6.033

Contents

1 Introduction

The Massachusetts Bay Transit Authority (MBTA) is seeking a design for a new real-time tracking system for its network of buses across the greater Boston area. This tracking system will enable the MBTA to collect data on several metrics by which it measures the success of its network, including availability, accessibility, reliability, and passenger comfort. The MBTA's current bus system already successfully meets its first two goals, but has historically failed to meet its goals for reliability and comfort. The MBTA has recently upgraded its infrastructure and is looking to improve on the latter two metrics through this real-time tracking system. The system proposed in this document utilizes the MBTA's existing buses, network, and central servers to evaluate and improve on the MBTA's success in meeting its specific goals for these metrics.

Our proposed tracking system design collects and processes bus and passenger data at scale, using this information to adapt to changing demand and traffic conditions. The design prioritizes system reliability, user experience, and modularity in a variety of common and uncommon scenarios.

This system aims to achieve the MBTA's specified goals for bus frequency, punctuality, and comfort to provide the best travel experience to the MBTA's passengers. Our proposed design uses the existing modules of the MBTA—buses, bus routes, radio network, and servers—in order to collect data relevant to the MBTA's goals, detect and respond to high demand or route issues, and obtain useful feedback from MBTA passengers. This proposal outlines our system design by describing each of these functions and their implementations, and specifying how these decisions impact the overall bus tracking system and the MBTA as a whole.

2 Data Storage

Within the MBTA warehouse, there are seven servers, each with an internal storage of 10 TB, which have a wired connection with a throughput of 1 GB per second between them. Each server or pair of servers has a specific function and stores the relevant data locally when possible, in order to improve the modularity of the system and minimize data transfer between machines. The servers are assigned as follows.

1. The radio server is connected to the radio transmitter and receiver and handles communication between active buses and the servers in the warehouse.
2. The route planning server stores the MBTA's static data sets and handles scenarios where routes must be added or modified. This server stores the MBTA's census data, bus meta-data, route and schedule data, alternate stop data, and operator data, all of which are useful for addressing route or capacity failures.
3. The backup route planning server is identical to the route planning server and is able to take over its functionality in case of issues.
4. The failure-detection server is responsible for processing video data of bus occupancy and alerting the route planning server in the case of a failure.
5. The short-term data server stores information on frequency of service, bus punctuality, and estimates of passenger comfort over the past day. This server also stores the estimated likelihood that a passenger will get off the bus at a given stop.
6. The long-term data server stores records of specific failures to meet comfort or reliability goals over the previous two weeks, as well as averages for daily total passenger count, low-income passenger count, transfer rate, and the short-term data listed above.
7. The mobile app server runs the MBTA mobile app and stores passenger feedback and recent bus location data.

The server subsystem is designed to be highly modular, so each server is responsible for one aspect of the bus system. If one server fails, other functionality of the system is not affected. The communication server, however, is an exception, as only one server can be connected to the radio network, and all information transferred to or from buses relies on it. The modularity also ensures that the rate of data transfer between servers in real time is significantly less than the 1 GB/sec wired connection between them, so throughput is not a major concern.

3 Data Collection and Distribution

Our system collects and stores a variety of data from the buses. Each bus must measure and transmit data on location, timing, and the number of passengers to the central server. This data is needed to robustly detect failures and assess reliability and comfort standards, taking into account the potential inaccuracy of the collected data. Furthermore, the limited bandwidth on the trunk radio limits potential communication between the bus control and server, meaning that much of the communication must take place overnight in the warehouse.

3.1 Location Data

At the beginning of each day, each bus receives information from the radio server about the day's route via the in-warehouse wireless network. In addition to the usual route information, this also includes an estimate of the rate at which passengers tend to remain on the bus at each stop.

During normal bus operation, the bus control displays the name of the next three upcoming stops on the bus-operator interface. Meanwhile, it polls the GPS sensor to determine the bus's location. When the bus is reaches a stop, the bus control records the ID of the stop and the current timestamp in memory, and updates the interface to include the next three stops.

3.2 Passenger Data

At each stop, the bus control attempts to maintain an estimate of the number of passengers riding the bus. At each stop, this count is scaled down by the estimated passenger retention rate at this stop, and increases by one for each payment reported by the payment system. The bus control keeps track of the amount of time during which the estimated number of passengers is over the bus's capacity (1.4 times the number of seats). It also tracks the total number of payments and number of transfers reported by the payment system. After each stop, the bus control sends a "heartbeat" message to the server detailing the stop ID, current time and current estimate for passenger count.

Video data is used to improve the accuracy of the passenger count tracking system. To avoid overloading the radio network, the server continuously maintains a priority queue of the buses, and notifies each in turn that it should send video data. The server adjusts the frequency of video data collection to ensure that the network is not overloaded. Upon receiving this notification, the bus captures five frames of video data from the security cameras and transmits them to the server. The failure-detection server processes these and determines the number of passengers on the bus, sending this more accurate estimate back over the radio. The bus control updates its estimated passenger count to this value and stores a record of the accurate count. If the number of passengers in the video is greater than the capacity of the bus, the failure-detection server marks the bus internally as overcrowded. Keeping the passenger count accurate improves reliability, as failures can be detected more accurately.

Any time a bus not already known to be overcrowded sends a heartbeat message indicating that its estimated passenger count is over capacity, the server increases that bus's priority in the queue so that it will be accurately counted sooner. To ensure all buses are counted accurately, this reprioritization is only permitted to occur once for a given bus before it must wait through the entire queue again.

To improve the estimates of passenger retention rate, the bus control also occasionally captures and stores video data one stop later it transmits video data to the server. This video data is stored in the bus control along with the ID of the previous stop to be uploaded and analyzed once the bus returns to the warehouse. The bus control randomly chooses whether to perform this

storage every time video data is collected, ensuring not only that the videos do not overload the bus control's storage space, but also that information collection is spread throughout the day. At the end of the day, the bus control this video data to the server over the in-warehouse network. The failure-detection server then analyzes the video data to determine the difference in passenger count from before and after the stop, and sends it to the short-term data server. This data is averaged over time to improve estimates and is sent to the bus controls every day. Other statistics are also transmitted to the server at the end of day and aggregated: the time at which each stop was reached, the total number of passengers, the total number of transfers, and the total amount of time the bus estimated it was over capacity.

4 Route Planning

4.1 Initial Route Assignments

As part of its data storage, the MBTA servers maintain historical data on the usage of different routes. This data provides a baseline for the allocation of buses to routes that happens each day. In addition to this historical data, an employee for the MBTA will be in charge of adding descriptions of high traffic events to the bus system, such as a convention or Red Sox game. Thus, any planned events which create significant congestion can be approached as a normal day, though more buses will be necessary.

4.2 Failure Detection

As the buses go about their routes, they keep a running estimate of the number of passengers onboard, occasionally updating this based on information from the central server. If a certain percentage of the buses on a route are marked as overcrowded, then the failure-detection server notifies the route planning server of a need to add buses to the route. In the cases of a route becoming unavailable or a new route being added, MBTA administrators will be contacted directly, without requiring any work from our system to detect such a failure. In either case, the route planning server is sent the IDs of the buses and routes being modified, as well as the number of buses that will need to be mobilized.

4.3 Failure Recovery

If a new route is necessary for failure recovery, a system administrator will add the route to the list maintained by the server, along with a signal that it requires more buses. These new routes are either based on historically used routes (as in the case of subway line replacements) or designed based on the census and alternate stop datasets stored by the MBTA. Once the route is added, the server treats this as an issue of demand, as there is a route in demand with no active buses. The mobile app server is also notified that a new route was created, and sends an alert informing its users of the update.

When there is unexpected demand, the route-planning server is alerted by the failure-detection server. Once the system detects a route with insufficient buses, the server will add a bus to the route either by diverting a bus from another route or by sending a bus straight from the warehouse. If the route is close to the warehouse itself, then as many buses as necessary will be dispatched to the route. Otherwise, the server will move a bus from a nearby route. To achieve this, it first looks up nearby routes and finds the number of buses present there. Each route, along with a list of the buses currently on that route, stores the minimum number of necessary buses for normal operation.

Of the routes which have a number of buses above the minimum, the server selects the route with the greatest difference between its total passenger capacity and total estimated passenger count. The radio server sends the bus closest to the endpoint or midpoint of the selected route its new itinerary and route assignment, which are communicated to the bus operator by the bus control once the bus reaches the endpoint or midpoint. The route planning server then updates its bus assignment data set to indicate that this change has occurred.

Another type of failure occurs when a route needs to be modified in some way because certain stops or streets are inaccessible due to construction or an emergency situation. A system administrator is informed when this needs to happen, and they will use the census and alternate stop data stored on the servers to calculate a new route. Once this is calculated, the route is modified on the server, and both the buses on the relevant route and the mobile app server are notified of this change.

5 Mobile App

The proposed MBTA mobile app will serve an important method of communication between the MBTA and its regular passengers, and will supplement the MBTA's existing website. This app will provide its users with an estimate of when a bus will service a given stop, calculated based on bus location data and any bus reassignments made by the route-planning server. This app will also alert passengers of any unexpected route changes, and provide a feedback form through which passengers can submit ride ratings and specific complaints.

6 Use Cases

The MBTA bus system needs to operate effectively under three major use cases: normal use, high demand scenarios, and unexpected route failures. In any situation, if customers have complaints about a recent ride, this can be addressed using the convenient feedback form and corroborated by the recent failure records in the storage server.

6.1 Normal Conditions

Under normal conditions, the bus system meets all of its goals, as there are not enough passengers to exceed bus capacity, and traffic is light enough that buses reach all of their assigned stops in a timely fashion. Bus operators service their standard, most familiar route, and no failure correction is necessary.

6.2 High Demand

During periods of high demand, buses will be reallocated from the warehouse or less crowded routes to more crowded ones in order to maximize the average comfort of the passengers. Since only nearby buses are chosen to be rerouted, this system responds to failures quickly and keep bus operators near areas they know. A historic limiting factor has been a lack of idle buses, which our system alleviates by searching for buses from nearby routes rather than immediately using an idle bus.

6.3 Unexpected Route Issues

In the case of unexpected route issues, MBTA administrators use the census and alternate stop data, along with their experience maintaining the bus system, in order to calculate optimal route

changes. When a new route is designed, the system handles allocating buses to it in the same way as during a high-demand situation, meaning the new route becomes operational very quickly.

7 Conclusion

Our system is designed to meet the requirements and constraints inherent to the problem of providing real-time data collection, failure detection, and failure response for the MBTA bus network. By combining real-time data from the payment system and security cameras with historical data on passenger behavior, the system gains accurate passenger count estimates without exceeding the capabilities of the existing network infrastructure. Our system detects failures of high demand by analyzing these passenger counts and applying heuristics to determine when action is required, and responds to these failures by moving buses onto the overloaded route, either from the idle pool or from nearby routes as appropriate. In doing so, the system improves reliability and user experience for the passengers while remaining scalable and minimizing data transfer within the system The system's modularity also plays an important role in limiting this transfer, as all of the system modules store relevant and necessary data. By fulfilling these design goals, our system operates well under both normal conditions and conditions of high demand.

6.033 Computer System Engineering
Spring 2018