Big Brother: A reliable MBTA bus tracking system

Sourav Das, Tugsbayasgalan Manlaibaatar, Arsen Vasilyan

1. Introduction

MBTA has designed its bus system to be easily available and accessible to its users. Now, it needs a system that will manage the bus operation data. A system that processes these data serves a very important role in situations such as:

- A disadvantaged neighborhood of the city is consistently receiving less bus service than it needs.
- A bus route is having higher demand than usual for a special event like festivals or sport event.
- There is a road block for an accident or construction in a bus route and the buses on that route needs to find alternative paths.

Our system can respond to these difficult situations. It utilizes the infrastructure provided by MBTA in order to:

- Collect the data about the bus arrival times and the number of users.
- Process the collected data to store in the warehouse servers and keep statistics about the load on the system.
- Detect an unusually high load of a specific route and compensate for it.
- Adapt to changes in routes.

The system is capable of reacting to issues that occur at different timescales: from minutes to weeks. In most cases, it automatically responds to an issue. In few cases like a bus accident or route unavailability for instruction, it informs the information to the MBTA employee and delegates the responsibility to them.

While designing our system, we have made our design decisions according to the following priorities:

- Reliability and time constraints of bus arrivals.
- Ability to accommodate high demand spikes of the routes.

- Robustness and resiliency of the system components.
- User comfort.

Our system will help MBTA to provide its users a reliable and satisfactory service.

2. System design

2.1 Design overview

The system has four major components:

- **System monitoring component** collects information about the system. In particular, it collects data about the locations of all the buses, the number of passengers entering or leaving a bus at any of the stations. This data is consequently used by the other components of the system.
- **Statistics maintenance and analysis component** maintains statistics about bus usage. At the end of every day, it updates the statistics taking into account the new data it received from buses. It also analyzes the data it receives.
- **Bus reallocation component** detects the possible failures and reallocates buses accordingly. This component interacts with the statistics component to see if there are unusual situations that the system has to handle. Consequently, it makes the decision whether to reallocate buses from one route to another and whether to use recovery buses. It also handles the requests from MBTA to change routes or add new ones.
- **Passenger feedback and interaction component** interacts with the customers to get the satisfaction level of them and figure out possible improvements on our system. We will have a mobile friendly website that users can leave comments or questions.

2.2 Design Details

2.2.1 System monitoring component

There are two channels through which the bus control can communicate with the home server:

1. The wireless radio network: it is fast and can be accessed any time. The downside of using this network: it has a limited bandwidth.

2. The wireless network in the bus control: it has a high bandwidth. The downside of using this network: we have to store the data in bus control's memory. In addition, the server receives the data at the end of the day when all the buses arrive at the warehouse.

In our design, we use the first network to send the data that are critical for our system to our servers in real time:

- The GPS location of the bus is sent every half a minute.
- A notification is sent to the server every time the bus doors are opened or closed.
- The server is notified every time a passenger enters and pays for the ride using the pay terminal.
- 5 frames from the security cameras are taken between every two stops and the bus control sends them to the server in order to be analyzed using computer vision algorithms.

In the second channel, we communicate the data that are necessary for historical statistics purposes:

- Charlie card numbers, which the bus control receives every time a passenger uses a Charlie card to pay for the ride after entering the bus.

The most bandwidth-intensive data in this system is the video feed from the security camera. However, our assumption is that a bus will spend 3 or more minutes on average between two stops. If every bus sends 5 frames in every 3 minutes, it will only require approximately 1/3rd of all the radio bandwidth allocated for the buses.

To achieve a reliable system, the communication among the buses and the server is resilient to delays. To achieve tolerance to a sudden increase in latency, each bus control maintains a queue of the data that it sends to the server. The queue strategy helps to deal with a burst of demand for wireless communication from the buses that may cause latency issues.

Furthermore, the bus control switches from sending video feed to the data from the beam servers if the queue for the data is too large. This makes our system resilient to network malfunction and decreased throughput. Even though the accuracy decreases from 95-98% to 85-90% for the switch to beam server, it still operates in case of network failure.

Finally, this component robustly handles the situation in which we lose a signal from a bus. After missing 10 (or a different constant that can be chosen by system administrators) GPS notifications from the bus, it marks it as inactive and notifies the system administrators that it lost connection with the given bus.

2.2.2 Statistics maintenance and analysis component

First, this component runs the computer vision algorithms during the day on the security camera frames sent by the system monitoring component. It analyzes them and determines the number of people on each bus between any two stops. If the security camera data is not available, it uses the data from beam servers. Afterwards, it combines this data with the number of people entering the bus at every stop to infer the number of people leaving the bus at each stop.

Secondly, from the data received from the buses it calculates and maintains a historical record of the following quantities:

- Frequency of Service - calculated from GPS data
- Coverage - calculated from Census data and bus stop data
- Reliability - calculated from GPS data
- Comfort - calculated from the reported number of passengers at each bus
- Total Load - calculated from bus camera and Charlie card data
- Value to Network - calculate from Charlie card data

Additionally, it keeps statistics on the number of passengers on each route, as well as the average number of passengers entering and exiting each bus at every bus stop. It keeps them for every hour of day for every day of the week. We made the choice to organize the data this way, because we expect that the use of transportation is roughly the same every week. At the end of each day, it updates the average values using exponential decay averaging. We use following equation to update the values in the table:

$$X_{new} = \boldsymbol{\alpha} * X_{old} + (1-\boldsymbol{\alpha}) * X_{measure} .$$

The exact value for the constant $\boldsymbol{\alpha}$ depends on how much this data fluctuates. We can set it to be 0.9 in the beginning, but the system administrators of MBTA can change it if they think another value is more optimal.

Finally, we need to decide how long the system stores the historical information. This is dictated by the purposes for which we store the data:

- Investigation of the trends of use of the bus system by the customers in order to improve it.
- Troubleshooting our system, in order to address the feedback from the unsatisfied passengers.

Because of the yearly changes in the weather and the academic year, there should be trends on the timescale year, whereas for the troubleshooting purposes we probably would not need data older than a few months. Therefore, we find it reasonable to keep the data for 5 years. We estimate that storing it would take 100GB or less, assuming that the data store every day does not exceed 50MB.

2.2.3 Bus reallocation component

This component consists of two subcomponents:

- **Reallocation decision subcomponent** makes the decision whether a certain route needs more buses. Additionally, it handles requests from MBTA to add routes or to change a route in response for a part of it becoming unavailable.
- **Allocation subcomponent** makes the decision which bus to allocate to a route that needs more buses.

2.2.3.1 Reallocation decision subcomponent

Sometimes, we have to reallocate buses even if there is no failure because of high demand. We divide high demand into two stages - neutral stage, emergency stage (failure).

- Neutral stage - If for a specific bus the customer to seat ratio is more than one, we know that some of them might not be comfortable. Since the comfort of customers is one of the main priorities, we want to allow standing customers to have a seat. Therefore, the reallocation decision subcomponent sends a signal that the route requires an extra bus. In addition, it requests extra buses preemptively in case our statistics imply the route will have a high enough number of passengers.

- Emergency stage - if for a specific bus the customer to seat ratio is more than 1.4, we mark that route has an emergency demand. The execution subcomponent gets notified that the route is in the failure mode.

In addition, this subcomponent handles the other two possible failure scenarios:

- In case of adding an unexpected route, it adds a new route to the list of the routs and notifies the execution subcomponent to add new buses to it.
- In case of a part of the rout becoming unavailable, it requests the system administrators to provide it with the new route and modifies the route table to accommodate for that.

We made the decision to use MBTA employees to modify routes and not to do it automatically, because we believe that it happens rarely enough for it not to hurt performance and a lot of considerations such as passenger transit cannot be fully handled by our system. MBTA employees are also responsible to notify people of the route changes by putting signs at those bus stops.

2.2.3.2 Allocation subcomponent

This module finds the most appropriate bus to relocate to the given route. For each bus it calculates a cost function and chooses the bus with the least cost. This method is practical because we always have 1036 buses, so we do not need our system to be scalable. We have not yet decided on the exact form of the cost function, but it should express our priorities:

- Buses that will need less estimated time to arrive at the route have lower cost.
- Buses with operators familiar with the given route have lower cost.
- Add more?

In addition, we do not move buses on routes that have less than 75% of their buses and it does not use failure recovery buses unless it is notified of a failure.

Finally, this is the subsystem that sends each bus the list of bus stops that it's route will follow together with their GPS locations.

2.2.4 Passenger feedback and interaction

By having a mobile friendly website, we can allow customers to report their experience or ask questions of any kind both through their mobile and website. We will have one web admin who is responsible for filtering the comments from customers and answering questions. By doing so, we can reflect filtered customer feedbacks on our system design for further improvement

3. Conclusion

Our system design focuses modularity. Different modules of our system are responsible for the different requirements for the MBTA bus tracking system. Fault tolerance was one of our major design goals. We achieved it by incorporating mechanisms that would make our system resilient to failure of some of its components. For instance, the system handles gracefully loss of individual buses and malfunction in the radio network. Furthermore, modularity helps us to identify problems in the system in case of malfunction. Therefore, our system is able to work efficient and it is easier to find failure within the system. Furthermore, we also prioritized the comfort of our passengers, for instance by reallocating buses in case of a high but not yet critical passenger to seat ratio. Overall, the design of our system intends to optimize the experience of people.