DESIGN DOCUMENT

Problem Statement

Design and implementation of rerouting and rescheduling application for rapid mass-transit networks using artificial intelligence.

System Functions

Python Backend

Backend written in Python3. Core of the system. This includes AI algorithms that deal with rerouting and rescheduling. Also includes the algorithm for the fuzzy controllers. This will be running in real time, and will continue listening for any changes from the transit network. This also implements an administrative application that gives real time information to the administrator about the network as well as allows the administrator to “forcefully” close route(s). This would mean “forcefully executing” the rerouting and rescheduling algorithms according to the parameters input by the administrator. The results of this would finally be passed to the web-service.

Android front-end

To allow accessibility to a large userbase, an Android application will be deployed which will be the portal to access updates. The application will have a minimalistic layout, having search bars for querying source and destination stations and a button for querying the web service.

Upon clicking the search button, the app will send a GET request to the underlying web service using the Volley library to send the request. The response received will be a JSON object, with an array embedded within it having the processed results which will be displayed using a RecyclerView.

Django Web service

The entire back-end will be run on a web application created using Django, a framework written in Python. The service will accept requests from the clients, which here are the devices running the android application. Upon receiving a GET request from the clients, the service will parse the request and get the search parameters. The service will then call the python script which will calculate the possible trains, rank them and return this array to of results to the service which will send to the client this data enclosed as a JSON object.

NoSQL Database

A non-relational database used to store data such as station information, track information, and other various other factors that have the potential to influence the result of the rerouting-rescheduling algorithm. For this project, we have chosen ElasticSearch as our database which stores JSON files and uses TF-IDF to search

for results. Analogous to a relation in SQL, is an Index and similar to an entity is a Document. With this analogy in mind, the structure of the data is as follows:

There will be three indexes, Trains, Stations and Distances. Each document in trains index will contain train attributes like train id, starting station, destination, start time, and as a JSON array, the path to be traversed stored as a key value pairs having station code and duration to it. Station names will be aliased using station codes.

Stations will contain details about station like station name and station code. Distances document will contain the distance to each station from a station will also be stored as a matrix similar to an adjacency matrix.

TRAIN SAMPLE


```
{
        “id” : “123”,
        “starting_station” : “ABC”,
        “destination_station” :  “DEF”,
        “start_time” : 0700,
        “path” :  [
                        {“ABC” , 0},
                        {“XYZ_1” , 1},
                        {“XYZ_2” , 2},
                ]
}
```

STATION SAMPLE


```
{
    [
        {
            “station_name” :  “foo”,
            “station_code” : “BAR”
        },
        {
            “station_name” :  “foo”,
            “station_code” : “BAR”
        },
    ]
}
```


DISTANCE SAMPLE


```
{
    [
        “BAR” : [
            {
                “FOO_1” : 12
            },
            {
                “FOO_2” : 21
            }
        ]
    ]
}
```

Algorithms:

The system uses the following algorithms for various calculations:

1. Floyd-Warshall Algorithm
2. Kruskal's Algorithm
3. Fuzzy algorithm for indeterminate variables:
    - Fuzzification
    - Execute fuzzy rules
    - De-fuzzify
4. Coffmann-Graham Algorithm

Floyd-Warshall Algorithm

This algorithm will be used for calculating the shortest path whenever some situation causes some route(s) to be closed, allowing the network to function in spite of the problem.

Kruskal's Algorithm

This algorithm will be used for calculating the minimum time that would be required to traverse a route. This is will be used in tandem with the Floyd-Warshall algorithm to determine if the rerouting is possible, or if the train that transits on the route in question has to be shut down temporarily.

Fuzzy Algorithm

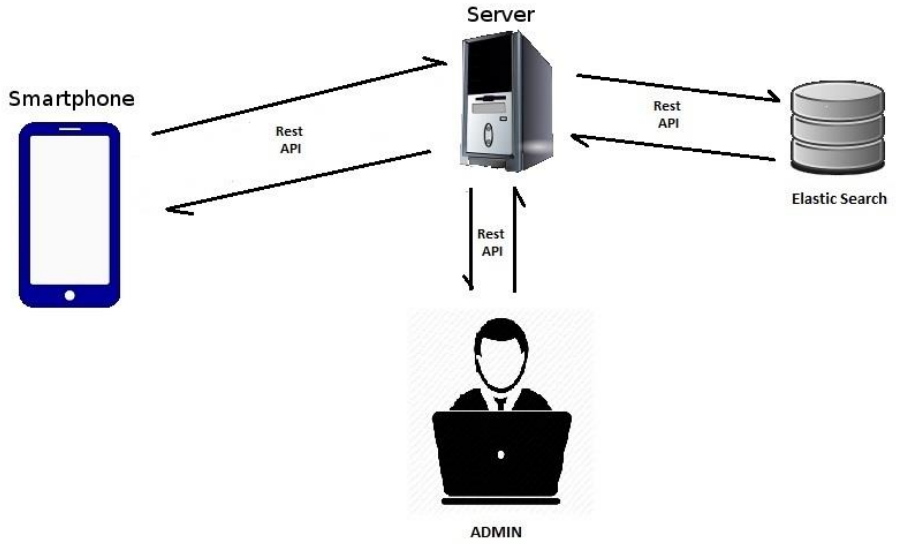
This algorithm will be used to take into account the indeterminate factors such as track conditions, traffic volume etc. These new fuzzified factors will in turn play a role in the deciding the new routes.

Coffmann-Graham Algorithm

This algorithm will be used for rescheduling the network after the rerouting is complete. The result is a new temporary schedule adjusted according to the result of the rerouting algorithm.

System Flow:

Overall System



System Backend (AI)