Mobility and Smart Cities

GILA - Greedy Incremental Louage's Algorithm

Presentation

Louage is a transportation system in Tunisia to overcome the lack of public transportation. This system is spread across all the Tunisian territory and is used by a lot of people. In this system their are two types of trips : regional and cross-regional.

This transportation system is very cheap and have a very high occupation level (97%).

Drivers propose their itineraries and stops with a time window of departure and arrival and a price per seat.

Passengers can ask to book a trip from a location to a destination with a desire departure and an arrival time window.

Operators assigned passengers to a trip if the trip is compatible with their time window and if there is enough seats available. First come first served (FIFO).

Other criteria could also be used like pricing, number of stops, number of changes require to achieve the trip etc.

Problem

The Louage probleme is a multi-constraints and multi-objectives problem.

Objectives :

- Maximize the number of passengers per vehicle (High occupation rate) to maximize the profit of the driver
- Maximize the number of satisfied requests (e.g. maximize the number of clients)
- Respect first in first out (FIFO) rule
- Minimize the waiting time of the passengers
- Minimize the overall sum of running itineraries duration
- As soon as a louage is full, it should leave

Constraints :

- Coherence of time windows and travel time with normalization
- Louage' s lines constraints
- Itinerary constraints
- Stop position depencies respected between request and organization (set of journeys)
- Respect the order of status and conditions on state transition for each journey
- Capacity of the vehicle
- Price

How can we assigned travelers to their corresponding while respecting the constraints and maximizing the objectives ?

Conception

To solve this problem the GILA algorithm uses a greedy approach. We makes the best decision at each step to reach a local optimum.

Inputs

We have two types of input in different lists : demands and offers.

Demands

Each client create a demand with the following information :

- Itinerary / Path (origin, destination)
    - Contains all the stops of the itinerary
    - Time window of departure (hd_minus, hd_plus)
    - Time window of arrival (ha_minus, ha_plus)
- Booked placed (NP)
- Bill
- State (initialized, validated, paired, contractualized, running, realized, non realized and archived), this is used to keep track of the state of the demand during the algorithm.

Offers

Each driver realized a trip with the following information :

- Itinerary / Path (origin, destination)
    - Contains all the stops of the itinerary
    - Time window of departure (hd_minus, hd_plus)
    - Time window of arrivals (ha_minus, ha_plus)
- Number of available seats (PlacesL)
- The driver
- The vehicle
- Status

Output

As a result, the algorithm returns an organization of journeys composed of associated the demands and offers.
Each journey is composed of :

- an itinerary
- an offer
- a list of demands
- a state
- a maximal number of seats occupied by passengers at any step of the itinerary

- the number of available seats at any step of the itinerary

Normalization

When clients are affected to a trip, their time window of departure and arrival are updated to take into account the traveling time and the time window of departure and arrival of the louage.

We have an offer leaving between 8h and 8h30 for a trip that takes 30min and arriving between 8h to 9h30.

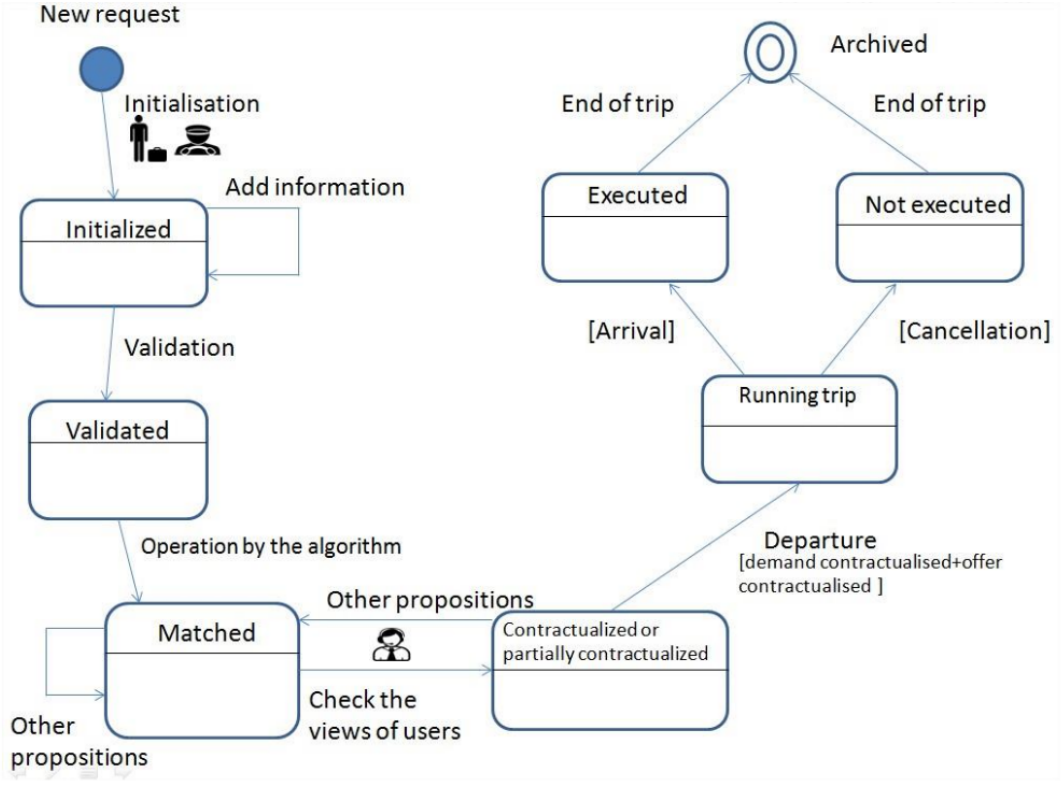
Normalization will shift the arrival time window of the offer to 8h30 and 9h30.

Trips must also be coherent:

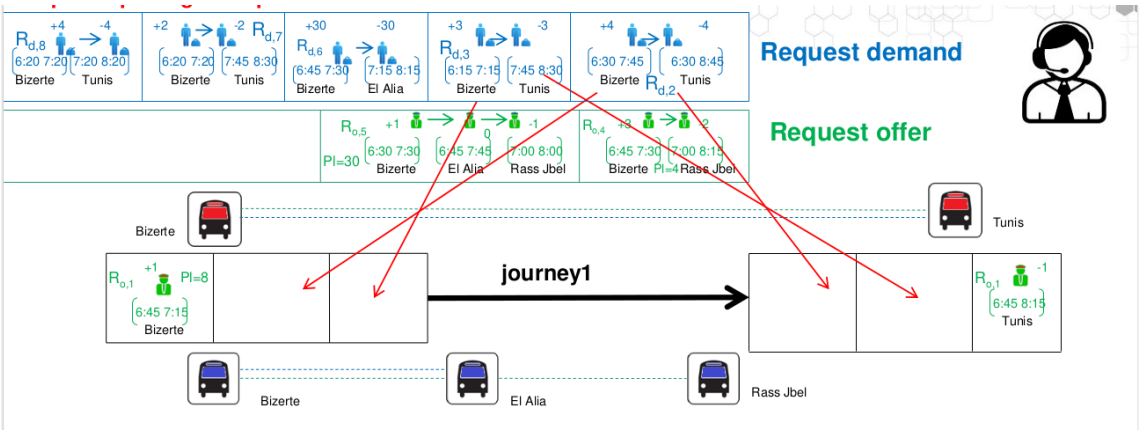
- each status of journey must be in the defined order
- it's impossible to drop someone before picking him up so pickup location have to be before dropoff location in the itinerary

Examples

Status



Pairing



Normalization