JTL
URBAN MOBILITY LAB AT MIT

SAV Simulation

Getting Started with `amod-abm` Platform

Jian Wen

September 14, 2017

SAV Simulation Platform Seminar
JTL: Urban Mobility Lab at MIT

Table of contents

Introduction: What's `amod-abm` and How It Works

Why amod-abm?

- **a**gent-**b**ased **m**odeling platform for **a**utonomous **m**obility-**o**n-**d**emand
- lightweight and fast:
  - a sayonara to AnyLogic
  - built-in routing server + flexibility in coding + FREE!
- `amod-abm` is good at:
  - studying vehicle operations and traveler behavior (as agents)
  - evaluating AMoD system designs such as fleet size, sharing policy, hailing strategy and pricing
  - testing operational models and dispatching algorithms

amod-abm on GitHub

- open-source repository at github.com/wenjian0202/amod-abm
  - do we want to pay for private repos?
  - or remove London and set up an imaginary case study area?
- platform for collaboration:
  - `README.md`: a brief tutorial
  - **clone**: copy the code and play with it
  - **fork**: copy the code and link with it, **fetch** new features and contribute, i.e., make **pull-requests** in the future
  - **issues**: where to report bugs

Features

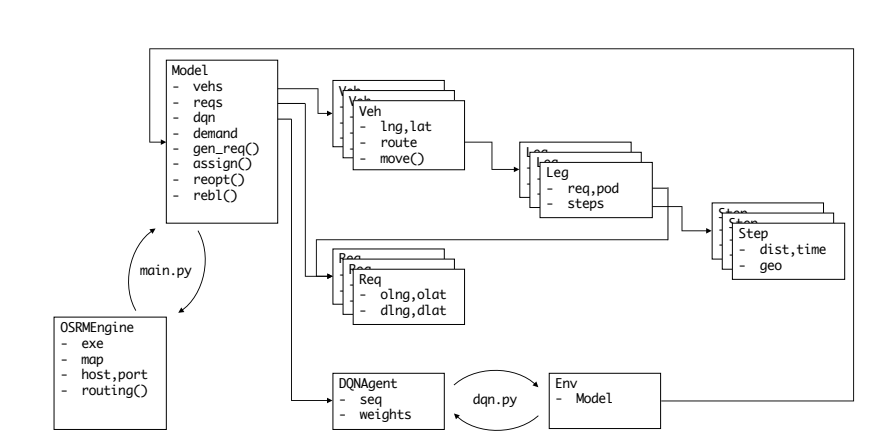
current features:

- free-floating system
- multiple algorithms for dispatching
- flexible fleet size / vehicle capacity
- time-invariant demand volume
- offline and static routing engine
- indicators: wait/travel time, detour, service rate and service/rebalancing distance, average load

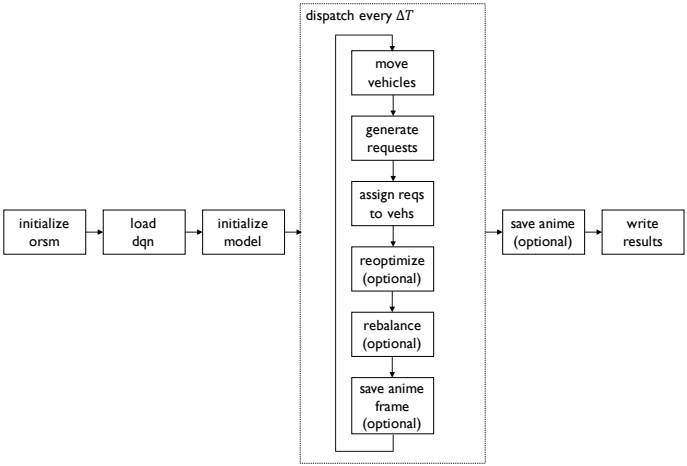
upcoming features:

- time-variant demand across a day
- automated interaction with demand prediction
- indicators: operational cost and revenue

Class Dependency



Workflow of `main.py`



Installation: Have `amod-abm` Working on Your Laptop

Prerequisites

- requirements:
  - OS X >= 10.10 with XCode (for Linux and Windows see here)
  - Python >= 3.6
- dependencies:
  - git
  - wget, boost, cmake, libzip, libstxxl, libxml2, lua, tbb, ccache, GDAL
  - we recommend using HomeBrew to install

Installation

- tutorial is found here
- key steps:
  - install all dependencies
  - get repo from GitHub by cloning or forking
  - recompile OSRM engine (written in C++, a compiled language)
  - download your map and extract it
  - launch the engine and give a try
  - run code!

Demonstration: Make Your Own `amod-abm` Project

`amod-abm` on GitHub

Code Demonstration

Play with Parameters

- parameters in `Constants.py`
- false `IS_ANIMATION` and `IS_ROAD_ENABLED` to speed up
- to play with:
  - fleet size `FLEET_SIZE` and vehicle capacity `VEH_CAPACITY`
  - demand matrix `DMD_MAT` and demand volume `DMD_VOL`
  - simulation time: `T_WARM_UP`, `T_STUDY` and `T_COOL_DOWN`
  - dispatching methods and their intervals: by default `MET_ASSIGN="ins"`, `MET_REOPT="no"`, `MET_REBL="orp"`
  - max detour `MAX_DETOUR` and max wait time `MAX_WAIT`

Make Your Own `amod-abm`

- new case study areas:
  - download and extract new map from Geofabrik
  - update demand matrix and volume in `Demand.py`
- implement new agents (e.g. bus)
- devise new algorithms
- pull request to contribute!

Thanks for listening.