Model Description & Problem Formulation

vwwong3

November 2018

1 MDP Model Description

The AMoD rebalancing problem can be modeled as a MDP with the following components:

Time step, t: In reality, it is more cost-effective to check on the status of all vehicles and do rebalancing regularly after a certain time period (e.g. 5 minutes). This time period can therefore be defined as a unit time 1, and each time the system checks on the status of the environment would be a time step $t \in \{0, 1, ..., T-1, T\}$, where T is the furthest time step that will be observed. At the start of each time step t, an agent starts out in a current state s_t, carries out an action a_t following a certain policy. At the end of this time step (or the beginning of time step $t+1$), a new state s_{t+1} and a reward r_t is observed.

Agent: Each vehicle is treated as an agent. Although this definition makes the problem a multiagent problem, it reduces the dimensions of actions, so that there is a smaller action space to explore.

State, s_t: The state at time step t is defined as a four-dimensional tuple, $(v_i, v_{it'}, \lambda_i, \lambda'_i)$.

The first two terms give information about all vehicles in the system. Consider a total of N stations in the environment. v_i represents the number of idle vehicles, or vehicles waiting at station $i \in N$ and not serving any customer at time t. v_i is an array of dimension i. $v_{it'}$ is the number of vehicles that will deterministically arrive at station i, and will become an idle vehicle at t', where $t' \in \{(t, T)\}$ (T, once again, is the maximum time that will be observed). This term is an array with a maximum dimension (worst case) of $min(M, N(T-t))$, where M is the total number of vehicles.

The last two terms give information about the customer demand. λ_i represents the number of customers waiting to get a vehicle at station i. λ'_i represents the demand forecasted with a LSTM neural network. The dimensions of these two terms are N.

With this state definition, each agent can observe the entire macroscopic environment.

[RI]:Can we concatenate your literature review and this? Also, please do start by describing the system. This paragraph should be used to describe the AMoD system qualitatively. What is it? what are its components? When do you get to act? What are your available actions?

[RI]:For example, this sentence seems like it already assumes a lot of background information.

[RI]:please define what's an agent first

[RI]:Interesting, wouldn't this mean that you have are essentially solving a distributed problem rather than a centralized one?

[RI]:This should should be earlier in the problem formulation, as part of the high-level description of the system under study.

[RI]:you mean, $|N|$

[RI]:Why? There

Action, a_t: An agent is only made to choose an action when it is idle. Consider an idle agent currently at station i. Its action is defined as j, where $j \in N$. If $i = j$, then the agent remains idle at its current station. If $i \neq j$, the agent departs to go to station j. The dimension of the action space is therefore N.

Reward, r_t: Is the average customer wait time.

Lastly, from classical MDP definitions, we define:
State-action values (expected utility if follow an action from state), $Q(s, a)$:
Values (expected utility), $V(s)$

2 Problem Formulation

2.1 Learning Stage

The goal in the learning stage is to learn the state-action values, $Q(s_t, a_t)$ at each state.

To estimate the optimal values, we will use the Q-learning algorithm:

for each (s_t, a_t, r_t, s_{t+1}) **do**
$Q_{target} = r_T + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$
$Q_{prediction} = Q(s_t, a_t) \leftarrow (1 - \alpha)Q_{predict} + \alpha Q_{target}$
end for

The algorithm approaches convergence as the following objective is met:

$$\min(Q_{target} - Q_{prediction})^2 \tag{1}$$

Notes: How will you deal with the fact that an action a_t will likely not have completed for several time steps? For example, if it takes K time steps to go perform action a_t, how will you account for $\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$ if a_{t+1} is empty for at least K steps?

[RI]:things to think about

2.2 Rebalancing Stage

At this stage, given the state-action values Q from the learning stage and the current state, we want to find actions that maximize the total state-action values, in order to achieve global optimal.

To do this, we formulate the following optimization problem. Consider a system of M vehicles in total, and each vehicle could be indexed as m.

$$\min - \sum_{m}^{M} Q_m((v_i, v_i t', \lambda_i, \lambda_i'), j) \tag{2}$$

[RI]:Looking at this, how will the vehicles coordinate if they are all choosing a locally optimal action?

where Q_m is the Q value function obtained in the learning stage, for vehicle/agent m.

The constraints are that inflow at all nodes must equal outflow:

- if there are $v_{it'}$ vehicles planning to arrive at station i at time t', then the number of idling vehicles at station i at time t' must be equal to $v_{it'}$
- the number of vehicles heading towards station i at time step t, must be equal to that plus the number of idling vehicles at station i at time step $t+1$

Formally, the constraints above can be written as:

$$\begin{aligned} &v_{it'}(t) = v_i(t'), \forall i \in N \\ &\textstyle\sum_{t'=t+1}^{T} v_{it'} + \sum_{m=1}^{M} 1(a_{t_m} 3 == j) = \sum_{t'=t+2}^{T} v_{it'} + v_j(t+1) \end{aligned} \tag{3}$$

[RI]:looking at your formulation, it is unclear to me why we would care about the constraints. The inflo-outflow constraints in the other papers are relevant because we are optimizing from a global perspective, thus our actions must be constrained to reflect where we do and do not have vehicles. In this case, each vehicle is optimizing individually, and the constraints are implicit in the range of possible actions.