Dynamic Pricing in Transportation Network Systems as a Markov Decision Process

Matt Battifarano

12-735 Urban Systems Modeling

7 May 2018

Outline

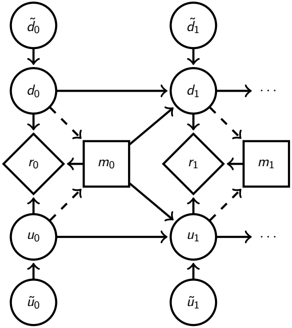
Introduction

- TNCs like Uber and Lyft use **surge pricing** to balance supply and demand.
- TNCs want to choose the surge multiplier which **maximizes revenue** given the supply and demand at a location.
- A **Markov Decision Process** can be used to determine the **optimal surge multiplier policy**.

Assumptions

- Single location
- The **state** at time t is given by the **number of users** and the **number of drivers**: $s_t = (u_t, d_t)$
- Users *always* accept a ride if there is no surge multiplier.
- Drivers *always* accept users.
- The **action** at time t is the **surge multiplier**.
- When the surge multiplier is **high**:
  - **Users** will be more likely to **wait**.
  - Some may decide to **leave.**
  - **Drivers** will be **attracted** to the location.
- **Users** and **drivers** arrive stochastically at the location with rates λ_u and $\lambda_d(m)$.
- User and driver choices are i.i.d.

The Model



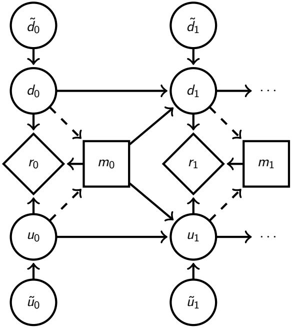
Variables:

- u_t: number of users
- d_t: number of drivers
- m_t: surge multiplier
- r_t: revenue

Transitions:

- $u_{t+1} = u_t^{(\text{remain})} + \tilde{u}_{t+1}$
- $d_{t+1} = d_t - d_t^{(\text{match})} + \tilde{d}_{t+1}$

The Model

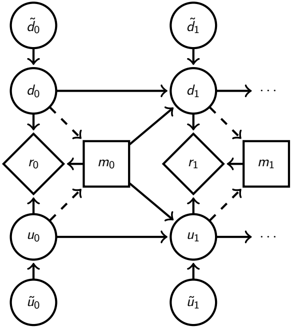


User Transition:

$u_{t+1} = u_t^{(\text{remain})} + \tilde{u}_{t+1}$

- $u_t^{(\text{accept})} \sim \text{Binom}(p(\text{accept} \mid m_t), u_t)$
- $u_t^{(\text{match})} = \min(u_t^{(\text{accept})}, d_t)$
- $u_t^{(\text{remain})} \sim \text{Binom}(p(\text{stay}), u_t - u_t^{(\text{match})})$
- $\tilde{u}_{t+1} \sim \text{Poisson}(\lambda_u)$

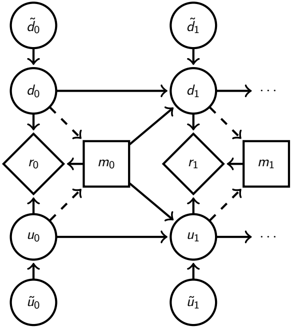
The Model



Driver Transition:
$d_{t+1} = d_t - d_t^{(\text{match})} + \tilde{d}_{t+1}$

- $d_t^{(\text{match})} = u_t^{(\text{match})}$
- $\tilde{d}_{t+1} \sim \text{Poisson}(\lambda_d(m_t))$

The Model



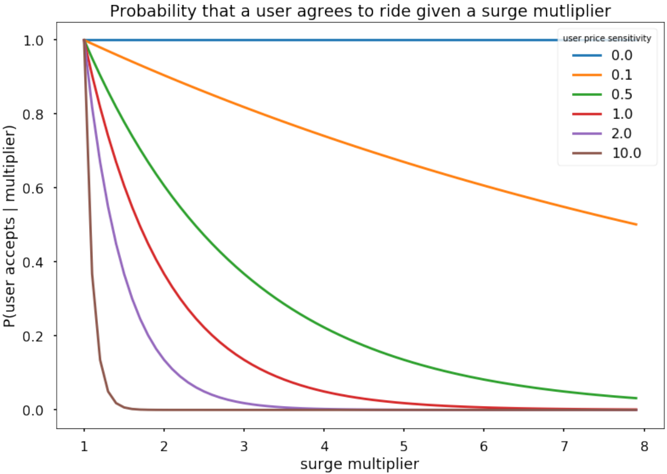
Revenue:
$r_t = R(s_t, m_t) = \text{fares} - \text{opportunity cost}$

- fares $= d_t^{(\text{match})} \cdot m_t \cdot$ base fare
- opportunity cost $= (d_t - d_t^{(\text{match})}) \cdot$ penalty
- Opportunity cost penalizes empty vehicles.

An Example Transition

What is the probability that a user decides to take a ride *after* seeing the multiplier?

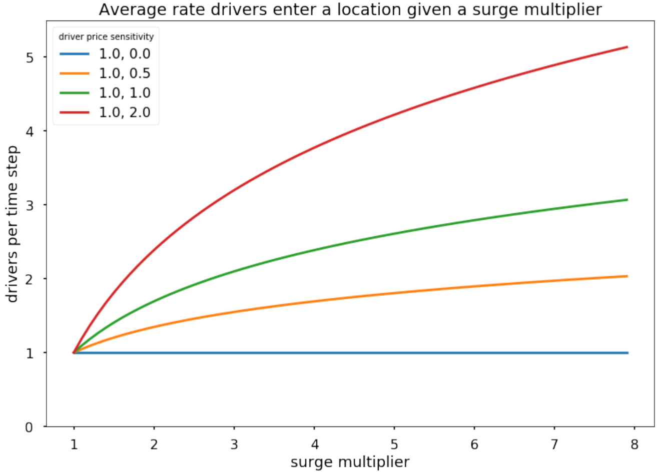
$$p(\text{accept} \mid m) = \exp(-\alpha(m-1))$$



An Example Transition

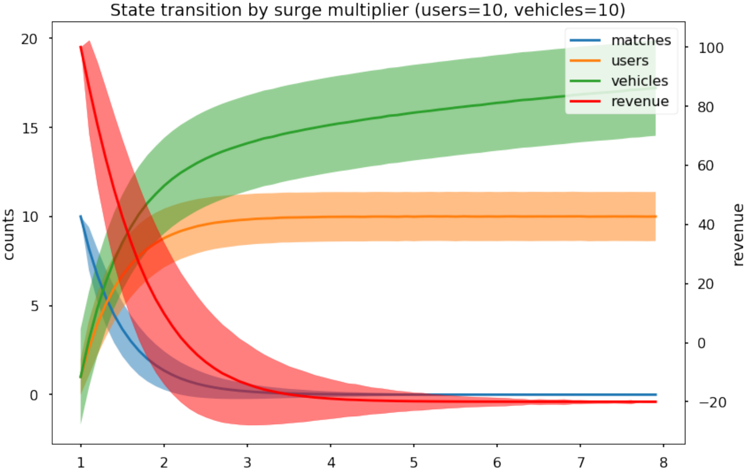
How many drivers do we expect to arrive to the location *after* seeing the multiplier?

$$\lambda_d(m) = \beta_1 \log(m) + \beta_0$$



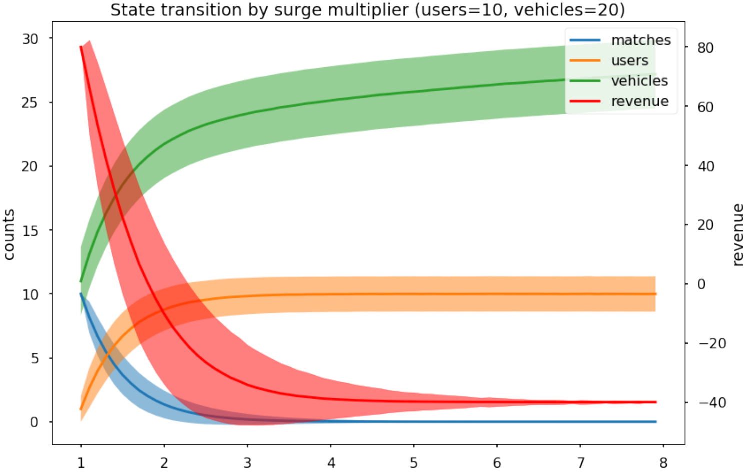
An Example Transition

What if drivers and riders **are balanced**?



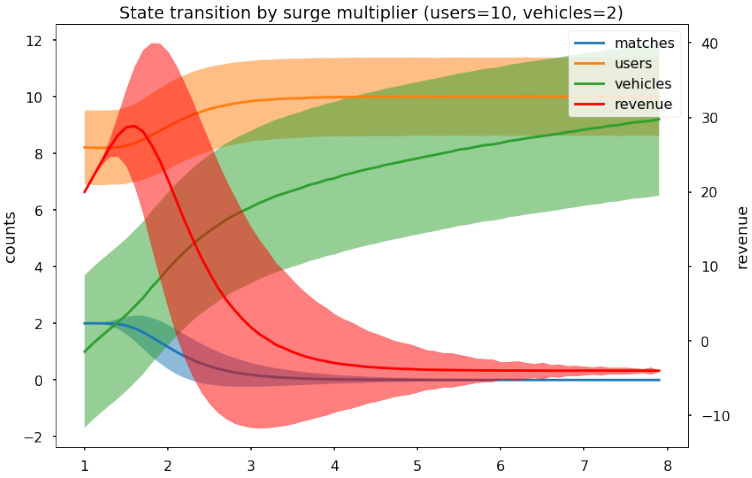
An Example Transition

What if drivers **are abundant**?



An Example Transition

What if drivers **are scarce**?



Finding the Optimal Policy

Value Iteration

- Initialize the value vector V_0
- For $k = 1, \cdots$
  - For each state s, compute the optimal policy:

    $$\pi_k(s) = \arg\max_m \mathbb{E}_{s' \sim p(s_{t+1}=s' | s_t=s, m_t=s)}[R(s_t, m_t) - \gamma V_k(s')]$$

  - For each state s, update V using pi_k

    $$V_{k+1}(s) = \mathbb{E}_{s' \sim p(s_{t+1}=s' | s_t=s, m_t=\pi_k(s))}[R(s_t, \pi_k(s)) - \gamma V_k(s')]$$

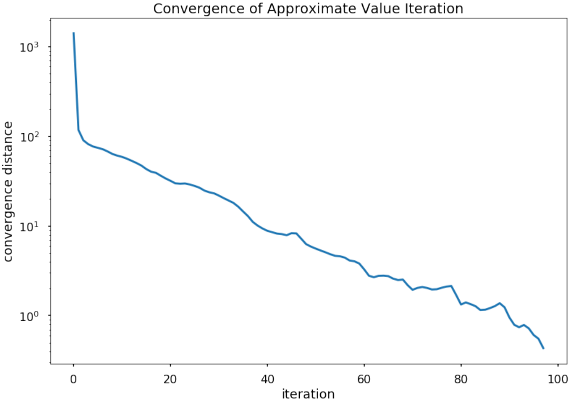
  - Until $V_{k+1} \approx V_k$.
- Return $\pi = \pi_{k+1}$

Finding the Optimal Policy

$$\mathbb{E}_{s' \sim p(s_{t+1}=s' | s_t=s, m_t=s)}[R(s_t, m_t) - \gamma V_k(s')]$$

- Problem: $p(s_{t+1} \mid s_t, m_t)$ is not known in closed form.
- Fortunately, it is **easy to sample from**.
- Idea: **approximate** the expectation with **Monte Carlo sampling**.

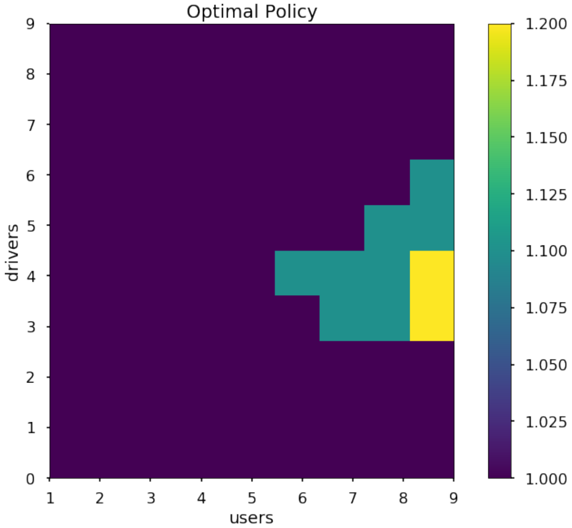
Results



Convergence of Approximate Value Iteration
convergence distance
iteration
10^3
10^2
10^1
10^0
0
20
40
60
80
100

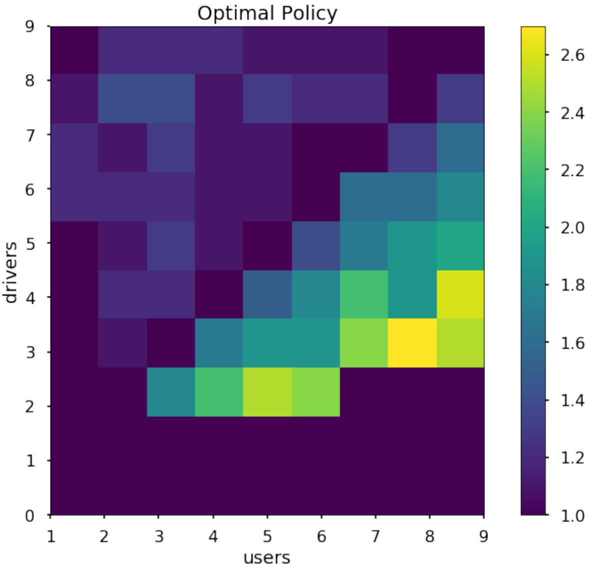
Results

What does the optimal policy look like when $\alpha = 2$?



Results

What about when users **care less** about surges? ($\alpha = 0.5$)



Future Work

- Make larger state spaces traceable: Approximate V_k and π_k using sampling and a **Gaussian Process**.
- Can we solve the **inverse problem**: given a sequence of surge multipliers can we infer the parameters $(\alpha, \beta_1, \beta_0)$?

Questions?

Thank you!
Code and Jupyter Notebook are available on GitHub:
`https://github.com/mbattifarano/surge-multiplier-mdp`