University of Michigan – Ann Arbor

Industrial & Operations Engineering Department

IOE 512 – Dynamic Programming

Design of a Parking Facility Pricing Strategy

A Markov Decision Process (MDP) Approach

Mouhamad Obeid moeobeid@umich.edu

Vinayak Bassi vbassi@umich.edu

Friday, November 16th, 2022

Abstract: The number of graduate students enrolling in the university gradually increases every year. The unavailability of price-efficient parking spaces around the university causes delays for students and has a negative socio-economic impact. It adds to a business opportunity to set up a parking space around the university which provides availability of parking spaces while enhancing social welfare. However, most of the parking stations have static price points throughout and thus, struggle with increasing demand. We propose to model a parking station activity as an MDP model to determine the optimal dynamic policy for each parking state. We define social welfare as a variable in terms of $. Our goal is to utilize the model to optimize social welfare while evaluating dynamic pricing policy. We use policy iteration and value iteration algorithms for infinite horizon MDP. Sensitivity analysis of both these algorithms was performed across various model parameters.

1. Introduction

With the advancement of the economy in the last few decades, the purchasing power of individuals has increased. Not only working employees, but even students are also able to afford a car to commute to university [1]. Consequently, parking slots have become quite an essential ground resource. A study conducted by D.C. Shoup et.al in 2006 for 11 cities across the world highlighted that searching for locating parking spots causes approximately 5000 additional miles of driving annually. This whole accounts for 3.78 million gallons of gasoline consumption since on average a driver will spend 8 minutes extra cruising for a parking spot [2]. This unavailability of parking resources at desired locations leads to a bad socio-economic life in the neighborhood. Apart from causing driver frustration, it also causes congestion and irrational driving around the area. With an increase in graduate students every year in the university this problem especially escalates in the areas near the university.

Another issue related to the parking lot is their existing pricing strategy. Currently, most of these parking lot uses static price points for daily operations. This causes two problems: firstly, with an increase in demand around the parking lot, the parking lot is generally full, and secondly, the pricing policy doesn't maximize the revenue of the parking lot operator and which future impacts the management of the facility.

Improvement in data collection techniques has enabled us to design dynamic settings for optimizing parking strategies. Dynamic decision-making enables one to know the availability of parking spots in the neighborhood. It also allows the facility to set up dynamic prices depending on the portion occupancy [3]. Therefore, policymakers can optimize their revenue by allocating prices in terms of incoming demand. Since the demand for the parking spot is dynamic in nature, it provides an opportunity to maximize revenue while maximizing utilization by using dynamic pricing schemes.

This study focuses on a case of a parking spot near the University of Michigan-Ann Arbor. The idea is to predict dynamic pricing strategies by modeling the parking station as a infinite horizon Markov Decision Process (MDP). We define a variable as societal welfare which is directly proportional to occupancy of the parking with a constraint of having a single parking spot available at any given time for emergency purposes. This variable is computed as a value function reward

for each state of the system and enables us to determine the pricing policy for that state. This study also briefly discusses the computational performance of Value iteration and Policy iteration algorithms used to optimize the dynamic setting across varied parameter settings.

2. Literature Review

There has been a lot of research work that has been published across conferences and journals that talk about the parking slot problem and its design. Junhao Huang et al. [1], laid out an optimal parking design to enhance the efficiency of parking occupancy and reduce congestion. They apply the Simulated Annealing (SA) general optimization design algorithm to determine the choice of the parking layout. The use of dynamic algorithms has been commonly adopted in much research. Zhendong Liu et. al [4] used an improvised form of the Dijkstra Algorithm to locate the best parking path and nearest free parking spaces. The work that formed the motivation for our project was based on the research work by Mirheli Amir et. al [3]. They developed a stochastic dynamic parking management model to minimize traveler's costs and maximize the parking agency's revenue simultaneously. They formulated the problem as a dynamic programming model with a stochastic look-ahead to determine parking price assignment using the Monte Carlo tree search algorithm.

Our research focuses on figuring out the optimal parking price policy for day operations that have deterministic demands. We formulate the parking place strategy as a Markov Decision Process and perform iterative algorithms to determine the optimal pricing strategy. We use a case of the University of Michigan to highlight how our model can be used to determine parking slot practices for effective social welfare for university students.

3. Modeling Framework

The next portion of the paper will discuss model formulation for the MDP model of the parking strategy problem. We will define *time horizons*, *state space*, *action space*, *transition probabilities*, and *rewards*.

Time Horizon: The key part of modeling an MDP is to decide whether the decision epochs are finite or infinite. Since we want to make decision-making based on states only, we formulate it as an infinite horizon MDP. The model is time-independent, and the absorbing state is considered to be the one when only 1 parking space is left.

State Space: We define a state space vector **S** which at every given decision epoch consists of non-negative integer values that represent the number of occupied parking spaces at the parking facility. For our model, we have taken a finite number of available parking slots are ten. This means that the maximum state value our system can be is 10. All the states are living states. We have no absorbing states. For each n parking slot, our **S** = {0, 1, 2, ...,n} where state 0 represents no occupancy, and state n represents full occupancy.

Action Space: We consider action space vector **A** consisting of non-negative integers that designate the parking slot price rates at each decision epoch n ∈ D. This action space will provide a suitable decision of prices at each state depending on the occupancy of the parking spot. We

define set $A(s_t) = \{1, 2, 3, ..., C\}$. Therefore, for every state $j \in S$, at each decision epoch $n \in D$, there will be a choice to set up ($a_t \in A(s_t)$) the parking rates till the next decision epoch.

Transition Probability: This is defined as the probability to transition from one living state to another living state given an action ($a_t \in A(s_t)$). For instance, the probability of the system transitioning to a higher parking occupancy state $j \in S$ from the current state $i \in S$ given the action a_t (i) is given by the transitional probability P_{ij}^{a}. Transition probabilities are generated using the python environment transitions method. This function gives out a transition probability matrix for a discrete event with input parameters such as state and action. These values come from an independent and identically distributed random distribution that specifies the fraction of time transition that will happen at each time step.

Rewards: Rewards are described as a real value that determines the parking preference at a given state and action pair. The reward value highlights the society welfare value for transitioning from state i to state j $\forall$ i, j $\in$ S. This value is also generated using the python environment's transition method and reward function. A set **R (s_t, a_t)** is defined which consists of rewards for each state and action pair. We assume all the rewards are given before the action is taken.

Solution Method: After defining the model framework for this parking problem, we attempted to find the optimal solution using two infinite horizon MDP algorithms: a) Policy Iteration, and b) Value Iteration algorithms. For each state, $s \in S$, we define a value function **V(s)** that represents the total expected dollar reward which represents the societal welfare component. The first step of numerical analysis is to conduct a policy evaluation for the existing policy of static price rates for parking. We define a bellman equation that is iterated for the given policy:

$$v(s) \leftarrow \sum_{a} \quad \pi(a|s) \sum_{s',r} \quad p(s', r|s, a)[r + \gamma v(s')]$$

Here, r refers to reward at state s, and for action a. γ represents the discount factor and $p(s', r|s, a)$ is the transition probability for switching from state, $s \in S$ to $s' \in S$, by acting $a \in A(s_t)$ at all timestep.

To determine the optimal policy, we perform Policy Iteration on this MDP. We start with using the same policy for which we conducted policy evaluation and then use a policy improvement algorithm at each state, $s \in S$ such that it modifies the policy at $s \in S$ to be greedy with respect to q-values generated from each value function for each policy.
We also perform the Value iteration algorithm on the MDP model to determine the optimal solution. The value function is iterated over a set of policies without maintaining a single policy explicitly until the value function converges. We iteratively update the following Bellman's optimality equation to find the value function for a set of decision rules/policies:

$$v(s) \leftarrow \max_{a} \sum_{s',r} \quad p(s', r|s, a)[r + \gamma v(s')]$$

4. Results

For numerical analysis of the model, we took a case of 20 slot parking facility near the University of Michigan. While defining the action space we took integer-valued parking prices from $0 to $7. We took the discount factor to be 0.9. Rest all the data considered for this analysis is arbitrally chosen, keeping in mind the rational ballpark.

We defined the initial policy to keep all the parking rates to be static at $1. When we conducted policy evaluation for this case, we can clearly see from the results in the Fig below, that the value function increases monotonically as more parking spaces are occupied until there is no parking space left.

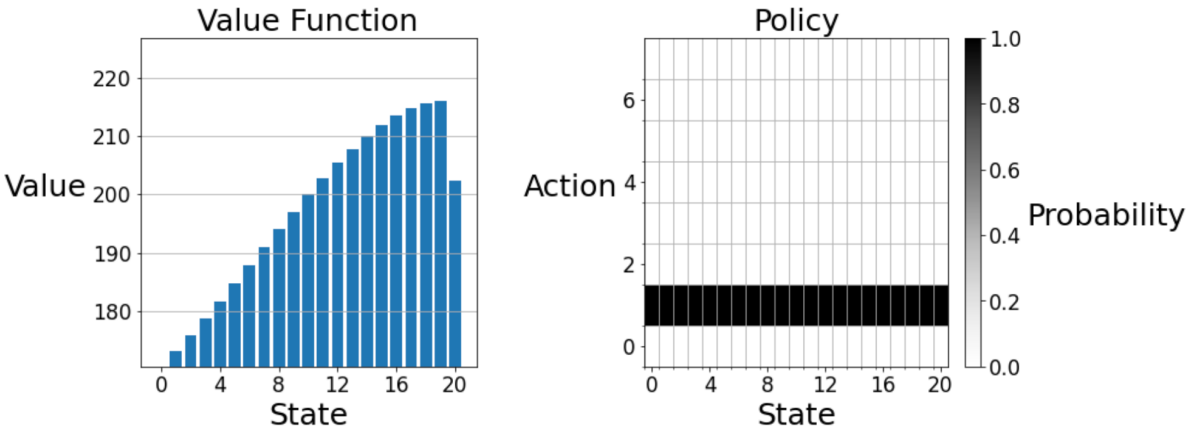

Figure 1 Profile Evaluation

Since the reward accrued is a simple function that maximizes in value when parking spots are taken except for the case of no parking available. For a dynamic set of policies, we were able to see changes in the value function and desired decision rules.

While performing the policy iteration algorithm, we started with the initial policy of having static parking prices at $1. However, we were able to get an optimum policy that depicted that until 16 parking spots are occupied, we will have no parking price but then at 17th state, the price should be kept at $1. It should be kept to $4 on state 18th. For the last two states, the price rate should be $7. In terms of total discounted reward received at each stage, we can see a monotonous increase till stage 16th whereas the maximum reward value is at state 19th accounting for $222. For the same price rate at full occupancy, the reward value drops close to $200.

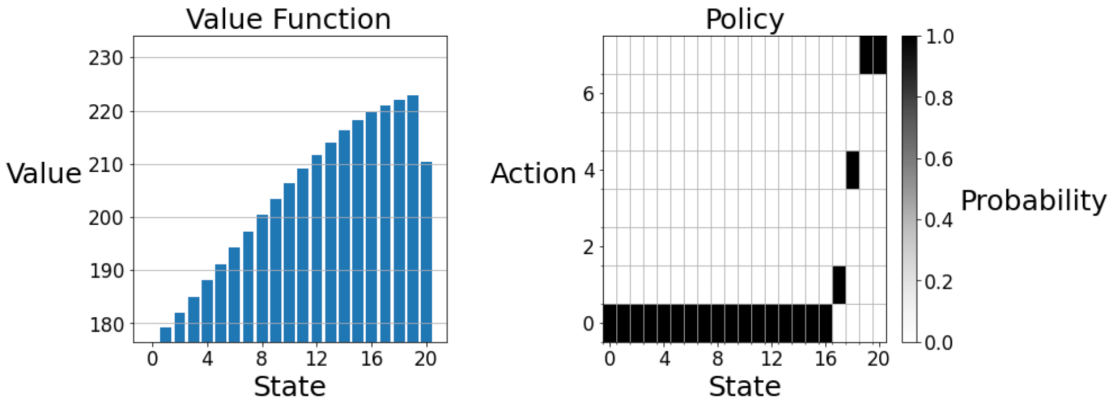

Figure 2 Policy Iteration

We were able to see the same results while solving for the value iteration algorithm.

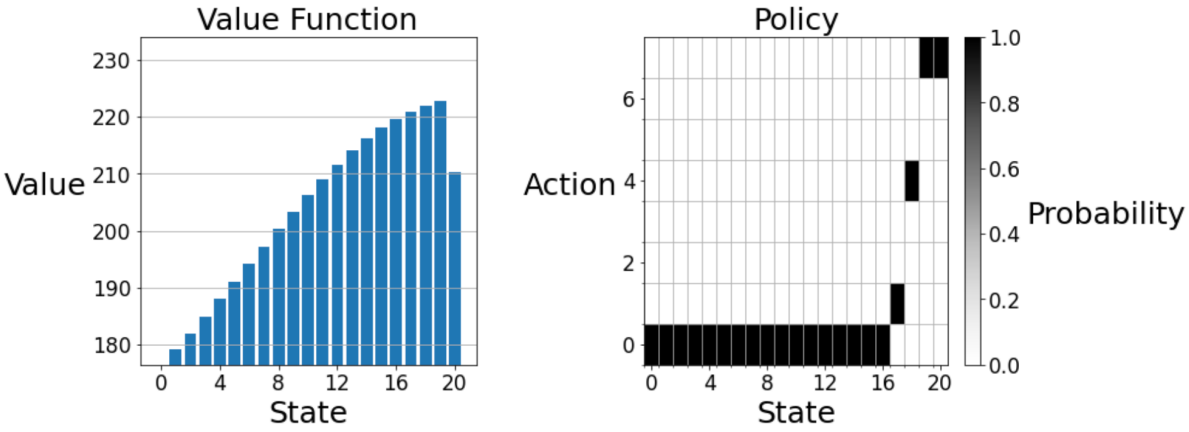

Figure 3 Value Iteration

However, in terms of computational performance, these two algorithms differ from each other. For the base case, we were able to see that the policy iteration algorithm was quicker than the value iteration algorithm. CPU time for the former was 1.12 sec as compared to the latter which was 1.92 seconds. We were able to achieve a quicker convergence when we changed the value of the discount factor from 0.9 to 0.75. The reported CPU time for the value iteration algorithm for the 0.75 discount factor was 700 ms. This trend of faster convergence was seen with a decrease in the value of the discount factor.

5. Conclusion

The parking strategy design problem is a common ground resource-related problem that can be solved using dynamic programming. We were able to formulate this problem as an infinite MDP while defining state space and action space. We checked the rewards associated with the existing

static policy and the trends were in terms of expectations. We used both policy iteration and value iteration algorithms to optimize the MDP and find the desired optimal policy. Results of both algorithms matched each other. However, since the data inputs, in terms of rewards and actions space used for our numerical analysis were simpler, therefore we can see a quicker convergence for the value iteration algorithm. Moreover, we also determined that the policy iteration algorithm is faster than the value iteration algorithm. The reduction of the value of the discount factor makes the algorithm converge faster. There is a potential of extending the model to stochastic modeling which will be part of future research.

6. Contribution

We both worked on referring to relevant literature and setting up the problem statement. Mouhamad Obeid worked on formulating the infinite horizon MDP model and solved it using the Policy Iteration algorithm. Vinayak Bassi worked on developing a python module called "Parking_Problem" which calculates transitional probabilities and reward functions for state-action pairs. He also worked on conducting numerical analysis with the Value iteration algorithm and conducting sensitivity analysis.

7. References

[1] Huang J., Liao T., Kang H., and Wang J. (2020). A General Algorithm for Rectangular Parking Optimization Design. Journal of Physics Conference Series. 1486 DOI: 10.1088/1742-6596/1486/3/032029.

[2] D. Ayala, O. Wolfson, B. Xu, B. DasGupta, and J. Lin, “Pricing of parking for congestion reduction,” in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst.*, Nov. 2012, pp. 43–51.

[3] A. Mirheli and L. Hajibabai, "Utilization Management and Pricing of Parking Facilities Under Uncertain Demand and User Decisions," in IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 5, pp. 2167-2179, May 2020, doi: 10.1109/TITS.2019.2916337.

[4] Z. Liu, Y. Yang, D. Li, X. Li, X. Lv and X. Chen, "Design and Implementation of the Optimization Algorithm in the Layout of Parking Lot Guidance," 2020 16th International Conference on Computational Intelligence and Security (CIS), 2020, pp. 144-148, doi: 10.1109/CIS52066.2020.00039.