DPRL Assignment 2: Optimal Policy for Deteriorating System
Group 63

Yilin Li
2737659
Vrije University Amsterdam
y45.li@student.vu.nl

Haohui Zhang
2722930
Vrije University Amsterdam
h17.zhang@student.vu.nl

1 Introduction

The target we intend to accomplish is to use Markov Reward Chains (MRC) and establish Poisson Equations to calculate long-run average costs and find optimal policy for deteriorating system problem. Specifically, we first propose to compute the stationary distribution and find long-run average replacement costs. Then, we intend to solve Poisson Equations to find an unique solution for the deteriorating system. Finally, we will consider preventive replacement policy for each state find optimal policy through utilizing policy and value iteration.

2 Explanation of problem and Building Markov Chains

Considering that a system start from "State 0" and the failure probability will increase from 0.1 (which increase every time unit linearly with 0.01), we propose to generate the MRCs which contains 91 different states. And each state is able to transfer to the first state ("State 0") and restart from it (**Figure 1**, top model). For example, the State $S(n)$ has a pass probability of $0.9-0.01n$ to transfer to State $S(n+1)$ and has a failure probability of $0.1+0.01n$ to transfer to State $S(0)$ with costs of 1.

The other way to design model is to settle a "Failure State". (**Figure 1**, bottom model) If the system come to failure, we will transfer the current state to "State Failure". Then, the system will transfer to "State 0" and this transition costs are 1. For example, the State $S(n)$ has a pass probability of $0.9-0.01n$ to transfer to State $S(n+1)$ and has a failure probability of $0.1+0.01n$ to transfer to State $S(F)$. At State $S(F)$, the only choice is to add replacement costs and transfer to $S(0)$. However, this model is not very suitable for this problem. We will illustrate more details in Conclusion.

3 Computing Stationary Distribution and Finding long-run Average Costs

Due to the deteriorating system problem can be abstracted as a standard Markov Chains, we realized that all formulas for MRCs can be used for solving problem. Therefore, the top model in **Figure 1** will be analyze and $\pi^T = \pi^T P$ will be used to compute the Stationary Distribution according to the **Formula [A.1]**, each state (from 0 to 90) can be expressed as:

$$\begin{cases} \pi(0) = 0.9\pi(1) + 0.1\pi(0) \\ \pi(1) = 0.89\pi(2) + 0.11\pi(0) \\ ... \\ \pi(n) = (0.9 - 0.01n)\pi(n+1) + \\ (0.1 + 0.01n)\pi(0) \\ ... \\ \pi(90) = \pi(0) \end{cases}$$

Which contains 91 different variables and infinitely solutions. Then, we intend to add **Formula [A.2]** into **[A.1]** to get an unique solution:

$$\sum_{i=0}^{90} \pi(i) = 1$$

Therefore, we get an unique solution of Stationary Distribution and intend to use **Formula [A.3]** to represent the long-run average replacement costs ϕ_*, which $r(i)$ can be defined as the costs of each transition (**Figure 1**, the last element in each "two-tuple"):

$$\phi_* = \sum_{i=0}^{90} \pi(i) r(i)$$

The result of long-run average replacement costs are: 0.146097. And **Figure 2** shows the stationary distribution from $\pi(0)$ to $\pi(90)$, which demonstrate the probability of system in state $S(n)$.

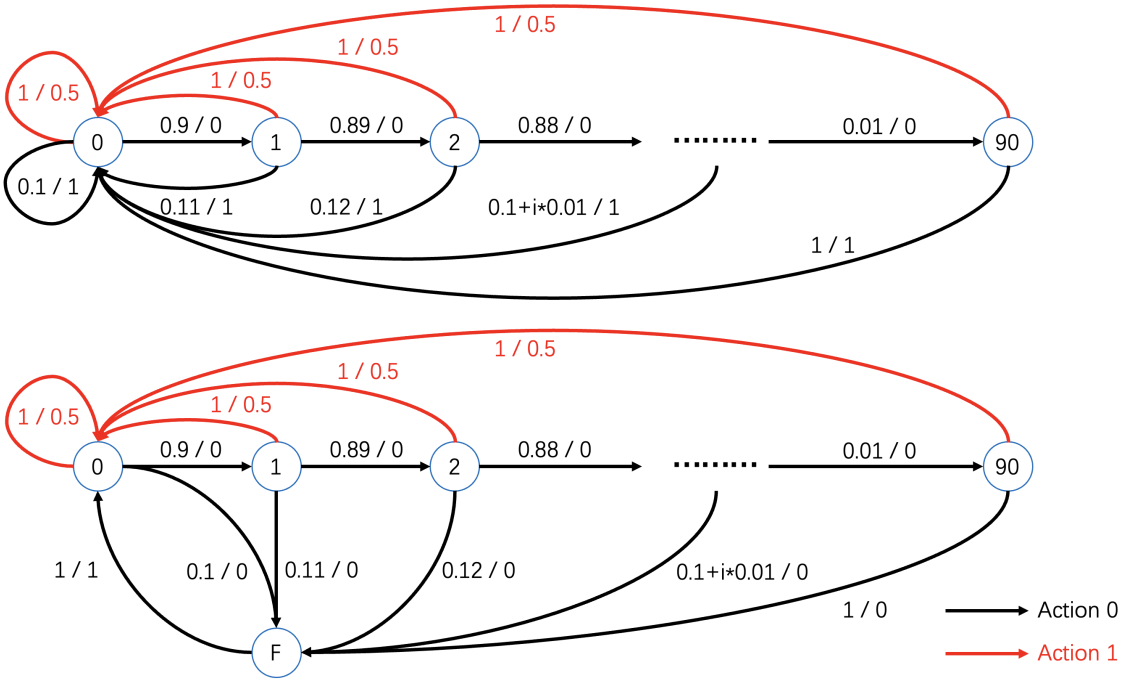

Figure 1: The top model shows the transition of Markov Chains without State F (Failure); The bottom model shows the condition after adding State F. Optimal Policy will be found through using policy iteration and value iteration.

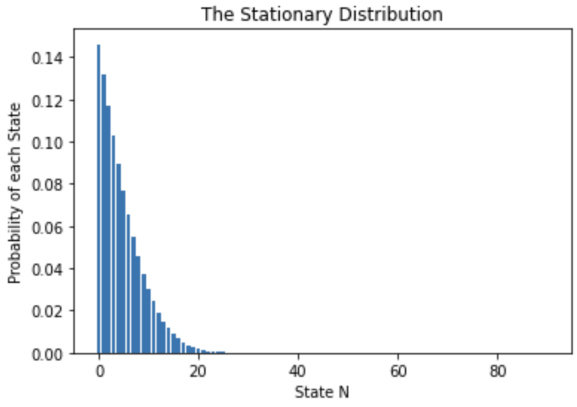

Figure 2: The probability of each state in Stationary Distribution. State 0 up to 14.6%

4 Solving Poisson Equations

According to the Poisson Equation $V_*(x) + \phi_* = r(x) + \sum_y p(y|x)V_*(y)$ and combining the question information, we can get 91 equations as **Formula [A.4]**:

$$\begin{cases} V(0) + \phi = r_0 + P_1(0)V(1) + P_0(0)V(0) \\ V(1) + \phi = r_1 + P_2(1)V(2) + P_0(1)V(0) \\ \dots \\ V(90) + \phi = r_{90} + P_0(90)V(0) \end{cases}$$

After plugging the probabilities into Poisson equations and transforming it, we rewrite the equations to solve the linear equation and get the **Formula [A.5]**:

$$\begin{cases} V(0) + \phi - P_1(0)V(1) + P_0(0)V(0) = r_0 \\ V(1) + \phi - P_2(1)V(2) + P_0(1)V(0) = r_1 \\ \dots \\ V(90) + \phi - P_0(90)V(0) = r_{90} \end{cases}$$

We get 91 equations and 92 variables (containing ϕ), so we need to suppose $V(0) = 0$ to get unique solution. After that, we can get one coefficient matrix which is 92*92 and a reward matrix which is 92*1, we use the numpy.linalg.solve function and find the total expected reward starting in $x = 0$ and the long-term average replacement costs ϕ. **Finally, the long-run average replacement costs which is calculated using Poisson Equations is 0.146097**.

5 Using Policy Iteration to Find Optimal Policy

In this section, new options are added to each state, which is at each state can have a preventive replacement with costs of 0.5 to go back to the first state. Thus, we need to make decision at each state to minimize the cost (**Figure 1, top model**).

Algorithm 1: Policy Iteration

Initialization:
Equations of Action 0 $P_0 = [92, 92]$; Equations of Action 1 $P_1 = [92, 92]$;
Fix an initial Policy $\alpha = [0] * 91$;
Reward for Action 0 $R_0 = [0.01 * i + 0.1] * 91$; Reward for Action 1 $R_1 = [0.5] * 91$
Begin:
Solve P_0 Through using α and R_0 ($V_0 + \phi_0 = r_0 + P_0 V$).
Record the solution of P_0 in $V[92]$.
while *True* **do**
  forall $i = 0, 1, ..., 90$ **do**
    (1) Compute $\alpha_i = argmin_{\alpha'}\{r_{\alpha'} + P_{\alpha'} V_i\}$;
    (2) $\alpha_*[i] = \alpha_i.index()$ (Select Action 0 or 1);
  end
  If α_* is equal to α ;
  Then break ;
  Establish Text Matrix $T = [92, 92]$;
  forall $i = 0, 1, ..., 90$ **do**
    (1) $flag = \alpha[i]$ (2) $T[i, :] = P_{flag}[i, :]$ (Establishing Equations)
  end
  Solve T ($V_n + \phi_n = r_n + P_n V$)., $\alpha = \alpha_*$
end
Output α

The target of iteration is to find the optimal policy, which means we need to choose the minimum $V+\phi$ from Action 0 and 1. **Algorithm 1** shows the method of implementing policy iteration. We first establish two-layers Poisson Equations based on the transition condition of Actions 0 and Action 1. After that, we assume that the initial policy of each state is all Action 0, and calculate the solution of this policy. In the iterative process, we bring the V_t from the previous moment into the equations for V_{t+1} and choose the smaller value. Which means Action N is more suitable for State $S(i)$ because Action N takes lower cost. This method will keep updating policies and values until the optimal policy does not change any more. **Figure 3** shows the optimal policy of iteration. According to the result, **State 13** is the first state which selects Action 1. And Action 0 are selected by State 0-12.

6 Using Value Iteration to Find Optimal Policy

In this section, we only compute V_{t+1} from V_t through using $V_{t+1} = max_\alpha\{r_\alpha + P_\alpha V_t\}$. **Algorithm 2** shows the whole process of value iteration and illustrate initialization and stopping criteria. **Figure 3** shows the optimal policy of value iteration. According to the result, **State 13** is the first state which selects Action 1 which is same as policy iteration. And Action 0 are selected by State 0-12.

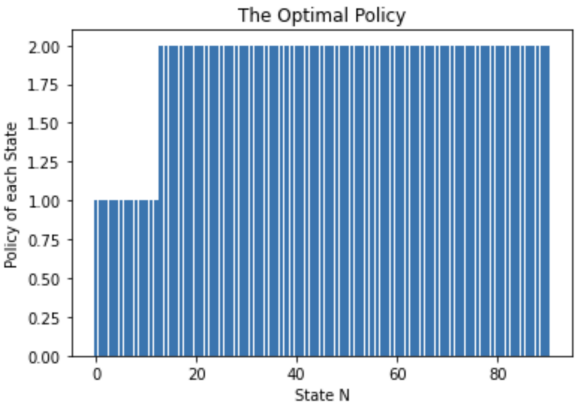

Figure 3: The optimal policy of each state. Start choosing Action 1 since State 13

In theory, $V_{t+1} - V_t \approx \phi$ will be used to determine the stopping condition of the value iteration. Then, the iteration will terminates with $(\phi_* - \epsilon)e \leq V_{t+1} - V_t \leq (\phi_* + \epsilon)e$. In actual operation, setting $(\phi_* + \epsilon)$ and $(\phi_* - \epsilon)$ are equal to $abs(V_{t+1} - V_t) = \delta(V)$. And $\delta(V) \approx$ **0.145** is sufficient to obtain the optimal policy. The algorithm executed for **32** times before stop.

Algorithm 2: Value Iteration

Initialization:
Equations of Action 0 $P_0 = [91, 91]$; Equations of Action 1 $P_1 = [91, 91]$; (without ϕ)
Fix a Policy $\alpha = [0] * 91$;
Reward for Action 0 $R_0 = [0.01 * i + 0.1] * 91$; Reward for Action 1 $R_1 = [0.5] * 91$
Value $V_t = [0] * 91$
while *True* **do**
  forall $i = 0, 1, ..., 90$ **do**
    forall $j = 0, 1$ **do**
      (1) Compute $V_{t+1}(i) = min_{\alpha'}\{r_j + P_j V_t(i)\}$ (Select the V_{t+1} with lower cost);
    end
    $\alpha[i] = argmin_{\alpha'}\{r_{\alpha'} + P_{\alpha'} V_i\}$ (Select Action 0 or 1);
  end
  Record V_{t+1} ;
  IF $(\phi_* - \epsilon)e \leq V_{t+1} - V_t \leq (\phi_* + \epsilon)e$;
  Then break ;
  $V_t = V_{t+1}$
end
Ouput α

7 Conclusion

We used Markov Chains and Poisson Equations to solve the deteriorating system problem. After analyzing the results, we believe that our results are in line with expectations. It can be seen from **Figure 2** that the proportions of State 0 is the greatest in the Stationary Distribution. And the proportions of other States decrease with time. From a practical point of view, this reflects that when time increase, the system is more likely to be damaged and enter State 0 to restart. At the same time, we also got the same ϕ in the process of solving Poisson Equations. In the iterative process using different strategies, both policy and value iteration got the same result. For each iteration, policy method need to solve equations and select optimal policy for each state while value iteration focus on calculating value for each state and clarifying the stop condition by using ϵ Since the State $S(n)$ with a larger n is more likely to be damaged, in order to minimize the cost, the optimal choice for these States are to implement preventive replacement.

Besides, we also implement and generate result by using the bottom model in **Figure 1**. However, the bottom model cannot fully describe the problem, but we still show the results. The long-run average cost is **0.127** and the plot of stationary distribution shows in **Figure 4**. **State 16** is the first state which selects Action 1. And Action 0 are selected by State 0-15 (**Figure 5**).

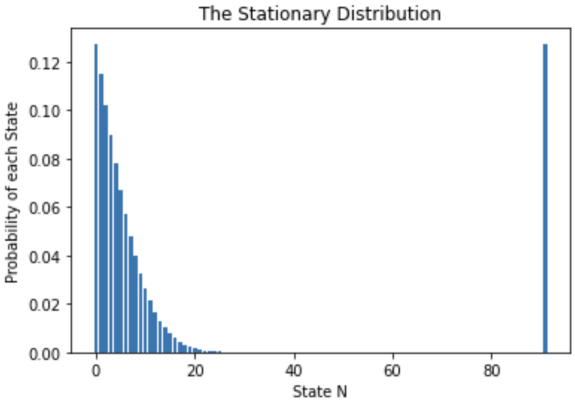

Figure 4: The stationary distribution of bottom model

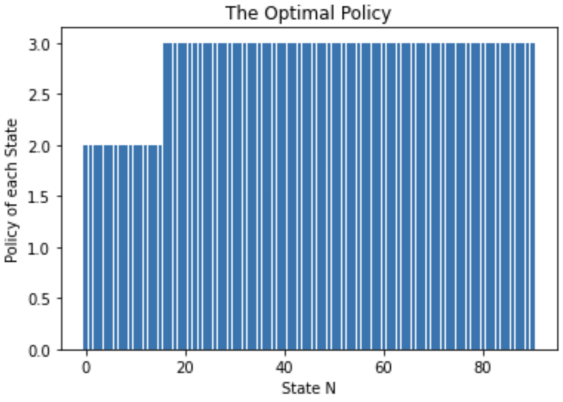

Figure 5: The optimal policy of bottom model