1 Model Formulation

Consider a system in which two infinite-capacity queues are attended by a single server. Customers arrive to lane i as a Poisson process with arrival rate λ_i. The server switches between both queues, so let the location of the server be denoted by $u \in \{1, 2\}$. Customers from queue i can only be served whenever $u = i$. We assume that switching and processing (service) times are deterministic, denoted by s and p, respectively, and that these actions cannot be preempted, once started. We model this system as a semi-Markov decision process, starting with a natural formulation and then showing how to derive a simplified version with a countable state space.

1.1 Natural formulation

The actions are *Process* (serve) a customer, *Switch* to the other queue and *Idle*, so the action space is $\mathcal{A} = \{P, S, I\}$. Let the number of customers in queue i be denoted by x_i. We will use the vector notation $x = (x_1, x_2)$ and define $e_1 = (1, 0), e_2 = (0, 1)$. Each state is identified by the tuple (u, ρ, σ, x), where ρ denotes the remaining service time and σ denotes the remaining switch time. The corresponding state space is

$$\mathcal{S} = \{1, 2\} \times \mathbb{R}^+ \times \mathbb{R}^+ \times \mathbb{N}^+ \times \mathbb{N}^+, \tag{1}$$

where $\mathbb{R}^+$ and $\mathbb{N}^+$ denote the nonnegative reals and integers, respectively.

We will first discuss the different types of state transitions that are possible, without exactly specifying the corresponding transition probabilities. For each transition, s denotes the current state, a denotes the action, s' denotes the next state and τ denotes the sojourn time. Let $\bar{u}$ denote the other queue. To support a concise description of the transitions, we set $\rho = p$ and $\sigma = s$ whenever the server is not processing or switching. Not every action is always available to the decision maker. Whenever $\rho < p$, the only allowed action is P and similarly, whenever $\sigma < s$, the only allowed action is S. Note that the server can only process a customer when the current queue is nonempty, hence action P is only allowed when $x_u > 0$. We have the following types of transitions:

- service completion ($x_u > 0$)

$$s = (u, \rho, s, x), a = P, s' = (u, p, s, x - e_u), \tau = \rho, \tag{2a}$$

- arrival to queue i when serving ($x_u > 0, \tau < \rho$)

$$s = (u, \rho, s, x), a = P, s' = (u, \rho - \tau, s, x + e_i), \tag{2b}$$

- switch completion

$$s = (u, p, \sigma, x), a = S, s' = (\bar{u}, p, s, x), \tau = \sigma, \tag{2c}$$

- arrival to queue i when switching ($\tau < \sigma$)

$$s = (u, p, \sigma, x), a = S, s' = (u, p, \sigma - \tau, x + e_i), \tag{2d}$$

- arrival to queue i when idling

$$s = (u, p, s, x), a = I, s' = (u, p, s, x + e_i). \tag{2e}$$

We will now discuss the rewards associated to the system. We assume that holding costs are levied at a constant unit rate for each customer in the system. Let Υ_m denote the sojourn time of the m-th transition and let $T_m = \sum_{j=1}^{m} \Upsilon_j$ denote the completion time of the m-th transition. The continuous reward rate process $\{r(t) : t \in \mathbb{R}^+\}$ is defined by

$$r(t) = -(x_1 + x_2) \text{ for } t \in [T_m, T_{m+1}), \tag{3}$$

where $m \geq 1$.

For optimization, we will consider an infinite horizon with the total discounted reward criterion

$$\phi_\beta = \mathbb{E}\left[\int_0^\infty e^{-\beta t} r(t) dt\right], \tag{4}$$

where $\beta > 0$ is known as the *discount factor*.

1.2 Reducing the state space

Although the above formulation is appealing because of its interpretability, we will now discuss how to obtain an equivalent model with a countable state space. We assumed that the serve and switch actions cannot be preempted. Therefore, no actual decision is necessary whenever an arrival occurs while the server is either serving or switching, so we skip these states and continue to the state where an actual decision must be made. This means that we disregard states with $\rho < p$ or $\sigma < s$ and the corresponding transitions in (2b) and (2d), so that states can now be represented as (u, x), with state space

$$\mathcal{S} = \{1, 2\} \times \mathbb{N}^+ \times \mathbb{N}^+. \tag{5}$$

Instead of modeling single arrivals, the transitions must now enable any number of arrivals to happen during a serve or switch action. In the following, let $N(t) = (N_1(t), N_2(t))$ denote a random vector with $N_i(t)$ being the number of arrivals to lane i that happen during a time interval of length t, which is distributed as a Pois($\lambda_i t$). The reduced model now has the following types of transitions:

- service completion ($x_u > 0$)

$$s = (u, x), a = P, s' = (u, x - e_u + N(p)), \tau = p, \tag{6a}$$

- switch completion

$$s = (u, x), a = S, s' = (\bar{u}, x + N(s)), \tau = s, \tag{6b}$$

- arrival to queue i when idling

$$s = (u, x), a = I, s' = (u, x + e_i). \tag{6c}$$

Note that the reward process $r(t)$ is still valid for the reduced model, but its value may change between our new states, which is not convenient if we want to apply existing methods that assume otherwise. Therefore, we will redefine the rewards in terms of an immediate reward (also called *lump sum* by some authors) that is earned at the start of the transition. Let $\{S_m : m \in \mathbb{N}^+\}$ denote the embedded state Markov chain of the natural model. Let $N \subset \mathbb{N}^+$ denote the indices of the states that are not skipped so that $\{S_n : n \in N\}$ is the embedded state Markov chain of the reduced model. Let $\bar{n}$ denote the predecessor of n in N, then the sojourn times in the reduced model are given by

$$\Upsilon'_n = \sum_{m=\bar{n}+1}^{n} \Upsilon_m. \tag{7}$$

By collecting the total reward over all skipped transitions, the immediate reward of the n-th transition in the reduced model is given by

$$R_n = \int_0^{\Upsilon'_n} e^{-\beta s} r(T_{\bar{n}} + s) ds. \tag{8}$$

Because we are now dealing with point masses at T_n for $n \in N \setminus \{0\}$, define

$$R(t) = \sum_{n \in N : T_{\bar{n}} \leq t} R_n. \tag{9}$$

Hence, we have

$$\int_0^\infty e^{-\beta t} dR(t) = \sum_{n \in N \setminus \{0\}} e^{-\beta T_{\bar{n}}} R_n \tag{10a}$$

$$= \sum_{n \in N \setminus \{0\}} \int_{T_{\bar{n}}}^{T_n} e^{-\beta t} r(t) dt \tag{10b}$$

$$= \int_0^\infty e^{-\beta t} r(t) dt, \tag{10c}$$

which shows that the discounted reward criterion (4) is exactly the same in the reduced model.

Because we will focus on the reduced model from now on, we let $\{S_m : m \in \mathbb{N}^+\}$ refer to the states of the reduced model and let Υ_m and R_m denote

the sojourn time and reward, respectively, of the m-th transition. Using $T_m = \sum_{j=1}^{m} \Upsilon_j$, the objective is given by

$$\phi_\beta = \mathbb{E}\left[\sum_{m=1}^{\infty} e^{-\beta T_{m-1}} R_m\right]. \tag{11}$$

2 Optimal Policies

We say that a policy π is *pure*, whenever the action at transition m is a function of the current state only. For the discounted total reward criterion, it can be shown that there exists a pure policy π^* that is optimal among all policies, i.e.,

$$\phi_\beta^{\pi^*}(s) = \sup_g \phi_\beta^g(s), \tag{12}$$

for some fixed initial state $s \in \mathcal{S}$. Well-known methods of finding optimal policies include policy iteration, value iteration and using linear programming.

2.1 Exhaustive service

A slightly more general SMDP model as introduced in Section 1 has been analyzed by Hofri and Ross (1987); instead of deterministic service and switch time, they consider random service times with the same distribution for both queues and switch times with possibly different distributions for both directions. They show (Proposition 2.1) that an optimal policy must always perform exhaustive service, which means that the server always continues serving the current queue when it is nonempty. In terms of our notation, this means that if $S_m = (u, x)$ satisfies $x_u > 0$, then the action $g^*(S_m) = P$ must be taken.

2.2 Double-threshold policy

For state $S_m = (u, x)$, we will use $g(S_m) = g(u, x_1, x_2)$ to simplify notation. A policy $g : \mathcal{S} \to \mathcal{A}$ is said to be a *double-threshold policy* if it performs exhaustive service, as defined above, and it satisfies

$$g(1, 0, x_2) = \begin{cases} I, & x_2 < m_2, \\ S, & x_2 \geq m_2, \end{cases} \tag{13}$$

$$g(1, x_1, 0) = \begin{cases} I, & x_1 < m_1, \\ S, & x_1 \geq m_1, \end{cases} \tag{14}$$

for nonnegative threshold values m_1 and m_2. It is conjectured by Hofri and Ross (1987) that there exists an optimal double-threshold policy (Theorem 3.4). Their proof is incomplete, but the missing part is reduced to two concavity properties of the optimization objective as function of the initial state. Our goal is to test their conjecture empirically by using model-free reinforcement learning to find an optimal policy (and estimate the true object function to verify the concavity properties).

2.3 Curse of modeling

In principle, it is possible to explicitly derive the transition probabilities of the model. However, this might turn out to be difficult in practice, a problem which is sometimes refered to as the curse of modeling. To avoid explicitly expressing the dynamics of the system, one could use simulation to obtain estimates of the transition probabilities, which can then be used in stochastic dynamic programming like value iteration or policy iteration to obtain an optimal policy for the approximation model. Alternatively, model-free reinforcement learning can be used, which circumvents the need to compute transition probabilities at all, which we will discuss next.

2.4 Q-learning

We will use the following notation for a general SMDP. Let $\pi : \mathcal{S} \to \mathcal{A}$ denote some pure policy. To avoid cluttering the notation, we ignore the dependency of the process on π. Let $\{S_m : m \in \mathbb{N}^+\}$ denote the embedded state process, then at each epoch m, some action $A_m = \pi(S_m)$ is taken, upon which reward R_{m+1} is observed and the state changes to S_{m+1} after Υ_{m+1} sojourn time. The transition probabilities are defined by

$$P_{xay} = \mathbb{P}(S_{m+1} = y \mid S_m = x, A_m = a). \tag{15}$$

Conditional on the event that the next state is y, we consider the joint distribution of the reward and sojourn time

$$F_{xay}(r, \tau) = \mathbb{P}(R_{m+1} \leq r, \Upsilon_{m+1} \leq \tau \mid S_m = x, A_m = a, S_{m+1} = y). \tag{16}$$

Note that we cannot simply consider the marginals in our case, because the reward and sojourn time are not independent. The continuous-time process $\{S(t), A(t), R(t) : t \geq 0\}$ where $S(t)$ is the state of the process at time t, $A(t)$ is the action taken at time t and $R(t)$ is the cumulative reward up to time t, is referred to as the SMDP.

When introducing the discounted total reward criterion in (4), we did not make explicit the dependency on the initial state s. Therefore, we define

$$\phi(s) = \phi_\beta^\pi(s) = \mathbb{E}\left[\int_0^\infty e^{-\beta t} dR(t) \;\middle|\; S(0) = s\right], \tag{17}$$

which allows us to easily define the Q-value of taking action a in state s as

$$Q(s, a) = Q_\beta^\pi(s, a) = \mathbb{E}\left[\phi_\beta^\pi(s) \;\middle|\; A(0) = a\right]. \tag{18}$$

Recall that the Q-update in classical Q-learning for MDPs with the discounted total reward criterion (see for instance Section 6.5 in Sutton and Barto (2018)) is given by

$$Q(S_m, A_m) \leftarrow (1 - \alpha) Q(S_m, A_m) + \alpha[R_{m+1} + \gamma \max_a Q(S_{m+1}, a)]. \tag{19}$$

In order to extend this method to be used with SMDPs, we need to take into account the fact that sojourn times affect the amount of discounting (Gosavi, 2015), so that the Q-update is now given by

$$Q(S_m, A_m) \leftarrow (1-\alpha)Q(S_m, A_m) + \alpha[R_{m+1} + e^{-\beta \Upsilon_{m+1}} \max_a Q(S_{m+1}, a)]. \tag{20}$$

3 Waiting

The action space that we have considered up till now is not complete in some sense. More precisely, given some feasible schedule y, there should be some policy π^y that depends in some (very complicated) way on y, such that executing π^y over the offline instance produces y. What we are missing is an action that allows the server to wait at the current queue for some finite time δ. Therefore, we define the additional waiting action $I(\delta)$ with $\delta \in \{\mathbb{R}^+, \infty\}$, such that $\mathcal{A} = \{P, S, I(\infty)\} \cup \{I(\delta) : \delta > 0\}$, where $I(\infty)$ is equivalent to the original I action. The $I(\delta)$ action causes the server to idle for δ time units at the current queue and then switch, unless another arrival happens at the current queue before δ time.

From the examples that we discussed in the offline scheduling setting, we conjecture that the optimal policy for the two-queue polling model must exploit some kind of waiting strategy. Therefore, we are going to evaluate double-threshold policies with waiting. In order to simplify the investigation, we consider a symmetric system in which the arrival rates $\lambda_i = \lambda$ are the same. Using a symmetry argument, we can show that this implies that $m_i = m$.

4 General Arrivals

We would like to drop the assumption of Poisson arrivals. Because the memoryless property does not hold anymore, we need to keep track of the time since the last arrival in the states. Therefore, states now need to be represented by the tuple (u, x, ν), where $\nu = (\nu_1, \nu_2)$ and ν_i is the time since the last arrival to queue i. We are now back in the situation of an uncountable state space, which requires us to apply discretization before being able to apply Q-learning.

5 Discussion

The idea of skipping states as we used in Section 1.2 is related to the use of *options* (Sutton et al., 1999), where the basic idea is that we can also choose *options*, which are a sort of sub-policies that specify how to take actions over multiple steps, in addition to ordinary or *primitive* actions. The original paper considers an MDP as the underlying model, but as they point out in a footnote (page 3), this idea can be extended to SMDPs (I have not yet searched the literature to see if someone has done this).

Most literature on reinforcement learning only focuses on expected quantities and ignores higher moments of the underlying distributions. In our model, the reward is not deterministic given the current state, action and next state (s, a, s'). I am not sure how this affects the convergence of Q-learning. The topic of distributional reinforcement learning (Bellemare et al., 2017, 2023) seems to be concerned with similar questions.

References

Marc G. Bellemare, Will Dabney, and Rémi Munos. A Distributional Perspective on Reinforcement Learning, July 2017.

Marc G. Bellemare, Will Dabney, and Mark Rowland. *Distributional Reinforcement Learning*. MIT Press, 2023.

Abhijit Gosavi. *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*, volume 55 of *Operations Research/Computer Science Interfaces Series*. Springer US, Boston, MA, 2015. ISBN 978-1-4899-7490-7 978-1-4899-7491-4. doi: 10.1007/978-1-4899-7491-4.

Micha Hofri and Keith W. Ross. On the Optimal Control of Two Queues with Server Setup Times and Its Analysis. *SIAM Journal on Computing*, 16(2): 399–420, April 1987. ISSN 0097-5397. doi: 10.1137/0216029.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning Series. The MIT Press, Cambridge, Massachusetts, second edition edition, 2018. ISBN 978-0-262-03924-6.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, August 1999. ISSN 00043702. doi: 10.1016/S0004-3702(99)00052-1.