Case study : Markov Decision Processes

Emmanuel Hyon[1]

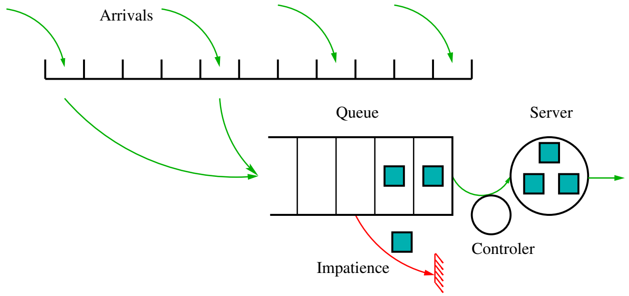
[1]Université Paris Nanterre
LIP6, Sorbonne Universités

RESCOM Summer School, June 2019

Outline

Outline

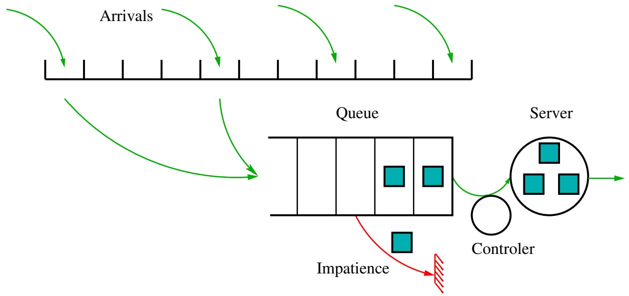
The Model



Arrival

- Customers arrive to an infinite-buffer queue.
- Time is discrete.
- The distribution of arrivals in each slot A_t, arbitrary with mean λ (customers/slot)

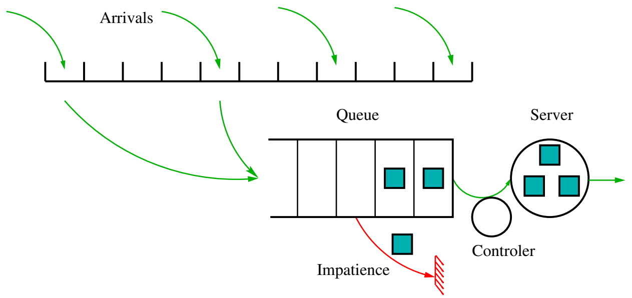
The Model



Services

- Service occurs by *batches* of size B.
- Service time is one slot.

The Model

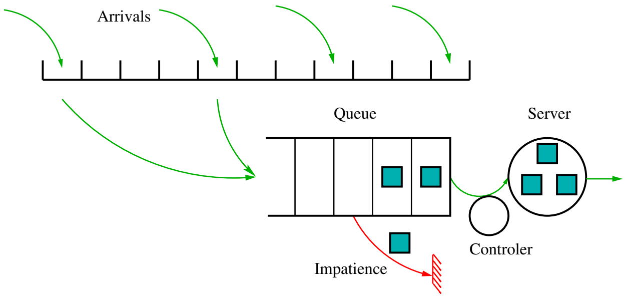


Deadline

Customers are *impatient*: they may leave before service.

- the individual probability of being impatient in each slot: α
- memoryless, geometrically distributed patience

The Model



Control

Service is *controlled*. The controller knows the number of customers but not their amount of patience: just the distribution.

The Question

What is the optimal *policy* π^* of the controller, so as to minimise the θ-*discounted* global cost:

$$v_\theta^\pi(x) \;=\; \mathbb{E}_x^\pi \left[\sum_{n=0}^{\infty} \theta^n \, c(x_n, q_n) \right] \;,$$

where:

- x_n: number of customers at step n;
- q_n: decision taken at step n;

and $c(x, q)$ is the cost incurred, involving:

- c_B: cost for serving a batch (*setup cost*)
- c_H: per capita *holding cost* of customers
- c_L: per capita *loss cost* of impatient customers.

Outline

The problem type depend on the available information.

We define a realisation as the value taken by the random value after a draw.

When all realisations are known

In this case we face up a scheduling problem *off-line*.
Selecting unitary task with release date and deadline.

$\Rightarrow$ Algorithm polynomial and Greedy Method [Chretienne]

Only the past realisation are known

In this case we face up a scheduling problem *on-line*.

- When we have no knowledge about the future tasks after the current slot, we face up an *on-line deterministic* problem.
- When we know only the distributions of the next tasks (customers) we face up a stochastic control problem or an on-line stochastic problem

Methods that use the expected values

Presentation

One replace the random value by its expectation value and one optimises on a deterministic problem.

Here this will be equivalent to have the same duration of impatiences and the same arrivals number by slot.

Disadvantage

We can be very far from the optimum.

For example :
We have 2 types of customers.

- with $\mathbb{P} = 1/2$ task T1 has duration $m/2$.
- with $\mathbb{P} = 1/2$ task T2 has duration $\frac{3}{2}m$.

But the average duration of the task is m !

Scenario Method Description

- We assume that we know the probabilities that a given number of realisations occur.
- We replace the realisation values (we build a scenario). We then obtain a deterministic problem that appears with a given probability that is known $\mathbb{P}_i$.
- We optimise this deterministic problem and we get a value V_i
- We obtain the global cost by an average computation

$$V = \sum_i \mathbb{P}_i V_i$$

Scenario Method Description

An example

For example : We know that

- with $\mathbb{P} = 1/3$ we have scenario 1
  - In slot 1 : we have 1 arrival, no lost
  - In slot 2 : we have 2 arrivals, 1 lost
  - In slot 3 : we have 1 arrival, no lost
- with $\mathbb{P} = 2/3$ we have scenario 2
  - In slot 1 : we have 4 arrivals, no lost
  - In slot 2 : we have 0 arrival, 3 lost
  - In slot 3 : we have 2 arrivals, 1 lists

Scenario Method Description (2)

Associated with scenario 1 we get a deterministic problem. We solve it and we obtain a value of the objective function V_1^*.

Similarly for scenario 2, we obtain V_2^*.

The global value of the problem is

$$V^* = \frac{1}{3} V_1^* + \frac{2}{3} V_2^*.$$

Disadvantages

Problem : Curse of dimensionality. Indeed, the more we want approach the optimal value the more we have scenarios.

Scenario Method Description (3)

Disadvantages (2)

We have a value but no action to take !

To find an action one has to decompose the behaviour and the actions in a decision tree.

An example (sub optimal) from van Heteynrick

1. With the solving we can get a decision.
2. On generate a sequence of scenarios.
   For each of them we record the optimal decision
3. We select one with respect of one criteria
   par ex. this one which appears the more.

Sample Path Analysis Analysis of trajectories samples

The path of a stochastic process

The path of a stochastic process $\mathcal{X}(t, \omega)$ is the set of the values it takes along the time.

An analysis of paths allows to make it possible to draw observation about the optimal policy.
Exchange arguments : We compare the trajectories which differ at different times with different action.

Advantage

We do not need to make computation (*stochastic order*).

Disadvantage

The passing from a property which is satisfied on a trajectory in a property which is satisfied in expectation is hard !.

Outline

State action and temporal decomposition

State:

x_n: number of customers in the queue at time n.

Action:

q_n is the decision at time n:
if $q_n = 1$ a service occurs,
otherwise $q_n = 0$ and no service takes place.

State action and temporal decomposition

Temporal Decomposition

1. Begining of slot.
2. Update the number of customers (instant operation):
   by counting the number of arrivals during the previous slot,
   by adding arrivals to the number of customers in the queue.
3. According to the number of present customers decide to launch a service or not.
4. Start of service.
5. Departures of the impatient waiting customers.
6. End of service (one time unit later).
7. Update the number of customers (instant operation):
   by taking into account departures.
8. End of the slot.

State dynamics equation

$S(x)$: the (random) number of "survivors" after impatience, out of x customers initially present.

$I(x)$: the number of impatient customers.
$\Longrightarrow$ binomially distributed random variables

A_{n+1}: the number of arrived customers during slot n.

System Dynamics Equation

We can define the Equation that describes the evolution of the state.

$$x_{n+1} \;=\; S\left([x_n - q_n B]^+\right) \;+\; A_{n+1}\;.$$

Transition probabilities

Following

$$x_{n+1} = S\left([x_n - q_n B]^+\right) + A_{n+1} .$$

We get the transition probabilities.
The transition probability from x when q is triggered, to arrive in y is:

$$\mathbb{P}\left(y|(x,q)\right) = \sum_{i=0}^{i=(x-qB)^+} \mathbb{P}(S=i)\mathbb{P}A(y-i)$$

Idea of proof

If the number of survivors is i, $(y-i)^+$ customers should arrives to reach y.

Costs

The cost at step n is:

$$c_B q_n \;+\; c_L I([x_n - q_n B]^+) \;+\; c_H [x_n - q_n B]^+$$

Immediate Cost

$$c(x,q) \;=\; q\,c_B + (c_L\,\alpha + c_H)\,(x - qB)^+ \;=\; q\,c_B + c_Q\,(x - qB)^+ \;.$$

Optimisation criterion:

$$v_\theta^\pi(x) \;=\; \mathbb{E}_x^\pi\left[\sum_{n=0}^{\infty} \theta^n\, c(x_n, q_n)\right] \;.$$

Dynamic programming equation

The optimal *value function* $V_\theta^*(x)$ is solution to:

The dynamic programming

$$V(x) = \min_{q \in \{0,1\}} \{ c_B q + c_Q [x - Bq]^+ + \theta \mathbb{E} \left(V(S([x - Bq]^+) + A) \right) \}.$$

Optimal Policy

The optimal policy π^* is stationnary *Markovian* and feedback: there exists a function of the state x, $d(x)$, such that

$$\pi^* = (d, d, \dots, d, \dots)$$

and $d(x)$ is given by:

The optimal policy

$$d(x) \ = \arg \min_{q \in \{0,1\}} \{c(x,q) + \theta \mathbb{E}\left(V(S([x - Bq]^+) + A)\right)\}.$$

Dynamic programming validity

We have unbounded costs so we can not use usual results but more specific ones.
The required assumptions are given in Theorem 6.11.3 in Puterman and the following properties should hold:
a) there exists a positive function on the state space, w, such that:

$$\sup_{(x,q)} \frac{|c(x,q)|}{w(x)} \; < \; +\infty \; , \tag{1}$$

$$\sup_{(x,q)} \frac{1}{w(x)} \sum_y p(y|x,q) w(y) \; < \; +\infty \; , \tag{2}$$

b) for every μ, $0 \leqslant \mu < 1$, there exists η, $0 \leqslant \eta < 1$ and some integer J, such that, for every J-tuple of Markov Deterministic decision rules $\pi = (d_1, \ldots, d_J)$, and every x,

$$\mu^J \sum_y P_\pi(y|x) w(y) \; \leqslant \; \eta w(x) \; . \tag{3}$$

Computation of the optimal policy

Since we exhibit the Bellman equation and all the elements we can compute the optimal policy.

It suffices to code an algorithm for the resolution !!

There were very few solvers of MDP and we had to compute the algorithm for each model.

But structural results give us the optimal policy without the need for a numerical computations.

Outline

Optimality Results for $B = 1$

Theorem

The optimal policy is of threshold type: there exists a ν such that $d(x) = 1_{\{x \geqslant \nu\}}$.

Theorem

Let ψ be the number defined by

$$\psi = c_B - \frac{c_Q}{1 - (1 - \alpha)\theta} .$$

Then,

1. *If $\psi > 0$, the optimal threshold is $\nu = +\infty$.*
2. *If $\psi < 0$, the optimal threshold is $\nu = 1$.*
3. *If $\psi = 0$, any threshold $\nu \geqslant 1$ gives the same value.*

Method of Proof : Structured policies

Framework: propagation of properties through the dynamic programming operator (*Puterman, Glasserman & Yao*).

Theorem (Puterman, Theorem 6.11.3)

Let V_w be a set of functions on the state space adequately chosen. Assume that:

0. $\forall\ v \in V_w, \exists\ d$, *Markov decision rule, such that* $Lv = L_d v$.

If, furthermore,

1. $v \in V^\sigma$ *implies* $Lv \in V^\sigma$,
2. $v \in V^\sigma$ *implies there exists a decision* d *such that* $d \in \mathcal{D}^\sigma \cap \arg\min_d L_d v$,
3. V^σ *is a closed by simple convergence.*

Then, there exists an optimal stationary policy $(d^*)^\infty$ *in* Π^σ *with* $d^* \in \arg\min_d L_d v$.

Method of Proof : One step

Property (Submodularity (Topkis, Glasserman & Yao, Puterman))

A function g is submodular if, for any $\overline{x} \geqslant \underline{x} \in \mathcal{X}$ and any $\overline{q} \geqslant \underline{q} \in \mathcal{Q}$:

$$g(\overline{x}, \overline{q}) - g(\underline{x}, \overline{q}) \leqslant g(\overline{x}, \underline{q}) - g(\underline{x}, \underline{q}).$$

Property (Monotone Control (Topkis, Glasserman & Yao, Puterman))

A control is said monotone if the function d $x \mapsto q$ is monotone.

Theorem

If $Tv(x, q)$ is submodular over $\mathbb{N} \times \mathcal{Q}$ then $x \mapsto \arg\min_q Tv(x, q)$ is increasing in x.

Propagation of structure

Theorem

Let, for any function v, $\tilde{v}(x) = \min_q Tv(x, q)$. Then:

1. *If v increasing, then $\tilde{v}$ increasing*
2. *If v increasing and convex then $\tilde{v}$ increasing convex*

Theorem

If v is increasing and convex, then $Tv(x, q)$ is submodular over $\mathbb{N} \times \mathcal{Q}$.

Optimal Threshold

The system under threshold ν evolves as:

$$x_{n+1} = R_\nu(x_n) := S\left([x_n - 1_{\{x \geqslant \nu\}}]^+\right) + A_{n+1} .$$

A direct computation gives:

$$V_\nu(x) = \frac{c_Q}{1-\theta(1-\alpha)} \left(x + \frac{\theta\lambda}{1-\theta}\right) + \psi\, \Phi(\nu, x)$$

$$\Phi(\nu, x) = \sum_{n=0}^{\infty} \theta^n \mathbb{P}(R_\nu^{(n)}(x) \geqslant \nu)$$

$$\psi = c_B - \frac{c_Q}{1-(1-\alpha)\theta} .$$

Optimal Threshold

The system under threshold ν evolves as:

$$x_{n+1} \;=\; R_\nu(x_n) \;:=\; S\left([x_n - 1_{\{x \geqslant \nu\}}]^+\right) \;+\; A_{n+1} \;.$$

We can deduce the following Lemma

Lemma

The function $\Phi(\nu, x)$ is decreasing in $\nu \geqslant 1$, for every x.

Outline

Related Literature

Optimal service control (without impatience):

K. Deb & R. Serfozo,
Optimal control of batch service queues.
Advances in Applied Probability, (1973).

K. Papadaki & W.B. Powell,
Exploiting structure in adaptive dynamic programming algorithms for a stochastic batch service problem.
European Journal of Operational Research (2002).

Optimal *admission* with impatience:

Y. Kocaga & A. Ward,
Admission Control for a Multi-Server Queue with Abandonment
Queueing Systems (2010).

Optimal control of service in presence of stochastic impatience:

E. Hyon & A. Jean-Marie
Scheduling Services in a Queuing System with Impatience and Setup Costs
The Computer Journal, 2012.