EL2805, Exercise 2: Modeling with MDPs

Robert Mattila

Today's central concepts

- Markov Decision Process (MDP)

  A Markov chain where the transition probabilities depend on the choice of an action. The goal is to find actions (a policy) that are optimal with respect to some objective function.

- State-space, S
- Actions
- Rewards (or costs): $r_t(s_t, a_t)$
- Transition probabilities: $P_t(s' \mid s, a)$
- Time-horizon and objective
  - Finite-horizon, $T < \infty$

    $$\mathbb{E}\left\{ \sum_{t=0}^{T-1} r_t(s_t, a_t) + r_T(s_T) \right\}$$

    Need to define the terminal rewards, $r_T(s)$.
  - Infinite horizon
    - Discounted, $\mathbb{E}\left\{ \sum_{t=0}^{\infty} \lambda^t r_t(s_t, a_t) \right\}$

      Two interpretations:

      i) Value of unit reward decreases with time at geometric rate λ

      ii) We are optimizing the total cost over a random time horizon. The system "shuts down" with probability $1-\lambda$ each time step.
    - Average reward, $\lim_{T \to \infty} \mathbb{E}\left\{ \frac{1}{T} \sum_{t=0}^{T} r_t(s_t, a_t) \right\}$

Algorithms to solve MDPs:

(More next session and in computer labs)

- Dynamic programming, backward induction for finite horizon:
  - $u_T^*(s_T) = r_T(s_T)$
  - $u_t^*(s_t) = \max_a \left\{ r_t(s_t, a) + \sum_{s' \in S} p_t(s' | s_t, a) \, u_{t+1}^*(s') \right\}$
  - $a_t^*(s_t) \in \arg\max_a \left\{ \text{———} \;\; '' \;\; \text{———} \right\}$
- Policy and value iteration
- Linear programming

Ex 2.1 b)

Try to pass N laws: $l_1, l_2, \ldots, l_N$.
Every law doubles initial wealth w_0.
Probability p_r of revolt $\longrightarrow$ lose everything.
Probability p_p of being rejected by parliament.
Can retire at any time. Want to maximize wealth. Model as MDP.

Solution:

We define the quantities that make up an MDP in order.

State-space:

Let

$$S = \{0, 1, \ldots, N\} \cup \{\text{Retired}\} \cup \{\text{Fired}\}.$$

The set $\{0, 1, \ldots, N\}$ of states denotes the number of laws that has been accepted.

The other two states represent if the dictator has retired or has been overthrown.

(Such states are called "auxiliary", "grave", "terminal" or "absorbing" states.)

Actions:

He can do two things at every time:

R - retire, or T - try to pass a law

Note:

The unit of time here is "number of times he has tried to pass a law", which is not necessarily equidistant in actual time.

(There could be one year when two laws are up for vote, and some year when only one is.)

reward/objective

Time-horizon and cost function:

There is a total of N laws, so $T = N$.
The finite-horizon total reward objective is appropriate:

$$\mathbb{E}\left\{ \sum_{k=0}^{T-1} r_k(s_k, a_k) + r_T(s_T) \right\}.$$

Rewards:

Since there is a risk that he loses his full wealth, we can model it as if he collects the wealth only when he retires.
(Otherwise, we would have to define a cost, i.e., negative reward, that zeros what he has accumulated up to the point that a revolt happens.)

Terminal rewards:

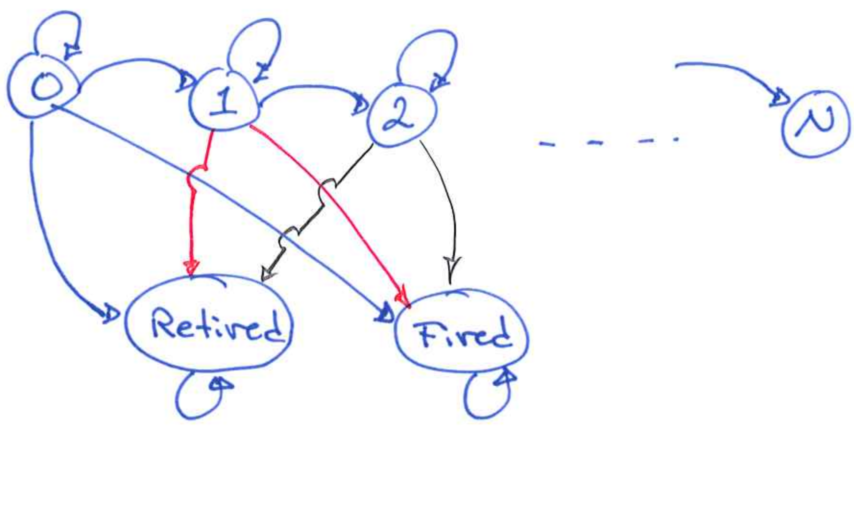
- $r_T(s=n) = \omega_0 \cdot 2^n$
- $r_T(s=\text{Fired}) = 0$
- $r_T(s=\text{Retired}) = 0$

Rewards:

- $r_k(s=n, a=R) = \omega_0 \cdot 2^n$ ← collect reward equal to initial wealth doubled as many times as number of passed laws when he retires.
- $r_k(s=\text{Fired}, a=\cdot) = 0$
- $r_k(s=\text{Retired}, a=\cdot) = 0$
- $r_k(s=n, a=\tau) = 0$

Transition probabilities:

It usually helps to draw the state-space to identify non-zero transition probabilities and only list those:



The non-zero transitions $p_t(s'|s,a)$ are:

- $P_t(s'=n+1 \mid s=n,\ a=\tau) = \underbrace{(1-p_r)}_{\text{no revolt}}\underbrace{(1-p_p)}_{\text{passed in parliament}}$

  that the law is passed successfully

  tries to pass a law

- $P_t(s'=n \mid s=n,\ a=\tau) = (1-p_r)\,p_p$

  it did not pass

- $P_t(s'=\text{Fired} \mid s=n,\ a=\tau) = p_r$

  there was a revolt

<u>Sanity check:</u>

The sum of all outgoing transitions from any state should be one:

under any action

$$\sum_{s'} P_t(s'|s=n,\ a=\tau) = (1-p_r)(1-p_p) + (1-p_r)p_p + p_r =$$

$$= 1 - p_p - p_r + p_p p_r + p_p - p_r p_p + p_r =$$

$$= 1 \quad \text{ok!}$$

- $P_t(s' = \text{Retired} \mid s = n, a = R) = 1$

  He decides to retire

- $P_t(s' = \text{Retired} \mid s = \text{Retired}, a = \cdot) = 1$

  He stays retired

- $P_t(s' = \text{Fired} \mid s = \text{Fired}, a = \cdot) = 1$

  He stays fired.

Ex 2.6 | The rational thief

He is caught with probability p, and if so, loses everything. Otherwise, he collects the valuables and adds them to his fortune. He can retire at any time.

Solution:

State-space:

We cannot take # of houses robbed (as we did in the previous exercise), since each house contains different valuables.

Let

$$S = \mathbb{R}_{\geq 0} \cup \{\text{Prison}\} \cup \{\text{Retired}\},$$

where is his accumulated fortune.

Actions:

Every night, he can do two things:

R - retire, or C - continue.

Time-horizon and objective:

There are multiple ways of modeling this problem. Let's first model it under the assumption that he is only fit to rob houses up to some age.

In this case, we have a finite horizon problem:

$$\mathbb{E}\left\{ \sum_{t=0}^{T-1} r_t(s_t, a_t) + r_T(s_T) \right\},$$

where T is for how many more days he is fit.

Rewards:

We let the thief actually acquire his earnings when he chooses to retire:

- $r_t(s = x, a = R) = x$
- $r_t(s = x, a = C_i) = 0$
- $r_t(s = \text{Prison}, a = \cdot) = 0$
- $r_t(s = \text{Retired}, a = \cdot) = 0$

One interpretation of this is that he puts the valuables in some warehouse for storage. When he retires he sells everything and collects a monetary reward.

Terminal:

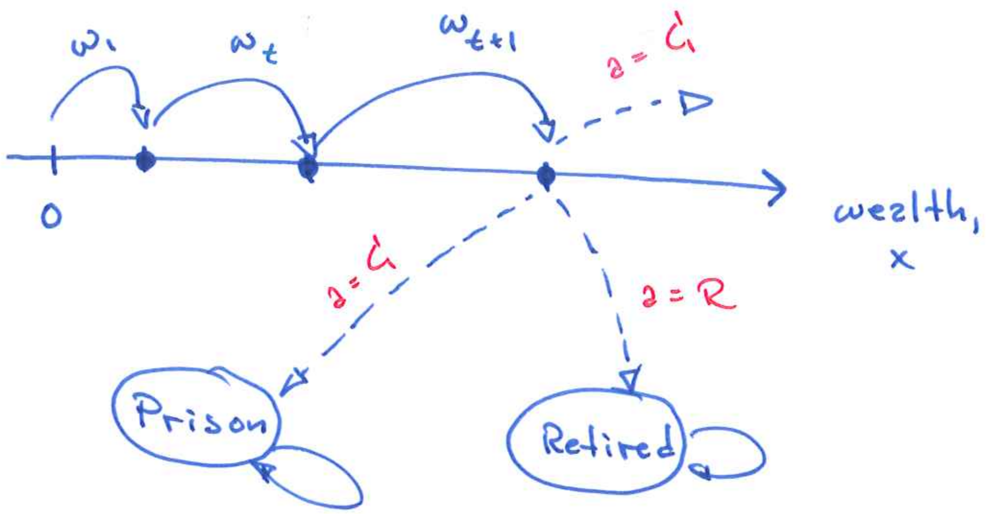
- $r_T(x) = x$
- $r_T(\text{Prison}) = 0$
- $r_T(\text{Retired}) = 0$

He acquires what he has accumulated, if he has not yet retired. (At this age, he is forced to retire.)

Transition probabilities:

Let the value of the valuables in the house robbed on night t be ω_t.

It's convenient to draw the state-space and only write the non-zero transitions:



- $P_t(s' = \text{Prison} \mid s = \text{Prison}, a = \cdot) = 1$
- $P_t(s' = \text{Retired} \mid s = \text{Retired}, a = \cdot) = 1$

He stays retired/in prison.

- $P_t(s' = \text{Retired} \mid s = x, a = R) = 1$ ← He chooses to retire
- $P_t(s' = \text{Prison} \mid s = x, a = C) = p$ ← He tried to rob a house, but got caught.
- $P_t(s' = y \mid s = x, a = C) = (1-p)\Pr\{\omega_t = y - x\}$

his new wealth

old wealth

he did not get caught

Probability that the house was worth $y-x$.

Assume pmf: TV, radio, etc.

Alternative interpretation:

There's nothing explicit in the exercise text regarding him only being fit up to some age. Essentially, as formulated, it is actually an ∞-horizon problem. How could we model the problem as such?

Remark (extra central concept):

The discounted ∞-horizon objective can be interpreted in two equivalent ways:

We optimize

$$\mathbb{E}\left\{\sum_{t=0}^{\infty} \lambda^t r_t(s_t, a_t)\right\},$$

i.e., we discount the future at a rate λ.

$\Longleftrightarrow$

We optimize

$$\mathbb{E}\left\{\sum_{t=0}^{T} r_t(s_t, a_t)\right\}$$

for $T \sim \text{geo}(\lambda)$, i.e.,

$$\Pr\{T = k\} = \lambda^{k-1}(1-\lambda).$$

system stays "on" for $k-1$ steps.

system goes "off".

Note:

$$\mathbb{E}\{T\} = \frac{1}{1-\lambda}.$$

That is, we are optimizing a total reward (i.e., undiscounted) criterion, but that the system "shuts down" with probability $1-\lambda$ in each time-step.

Solution (∞-horizon):

We remove the prison state, since it is equivalent to the process ending.

State-space:

$$S' = \mathbb{R}_{\geq 0} \cup \{\text{Retired}\}$$

Actions:

R - retire, C' - continue

Rewards:

- $r_t(s=x,\ a=R) = x$
- all other zero:

$$r_t(s=x,\ a=C') = 0$$

$$r_t(s=\text{Retired},\ a=\cdot) = 0$$

Time-horizon and objective:

Discounted infinite horizon:

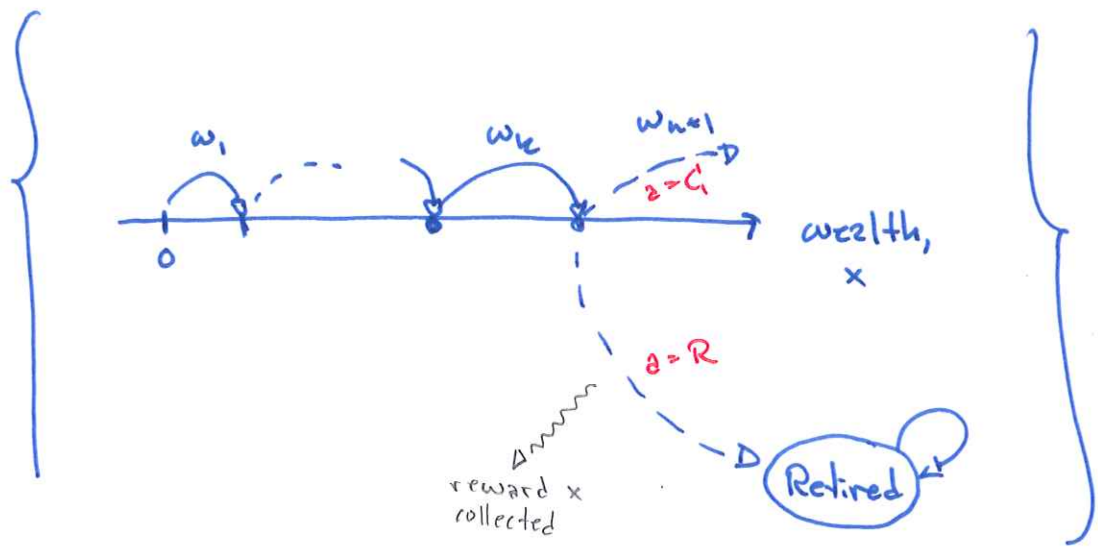
$$\mathbb{E}\left\{ \sum_{t=0}^{\infty}{}' (1-p)^t \, r_t(s_t, a_t) \right\}.$$

$\lambda = 1-p$ is the probability that the system does not "shut down"/terminate.

Transitions (non-zero):

- $P_t(s' = \text{Retired} \mid s = \text{Retired}, a = \cdot) = 1$
- $P_t(s' = \text{Retired} \mid s = x, a = R) = 1$
- $P_t(s' = y \mid s = x, a = C_1) = \Pr\{\omega_t = y - x\}$.

Interpretation:



with probability p of terminating at each time step.

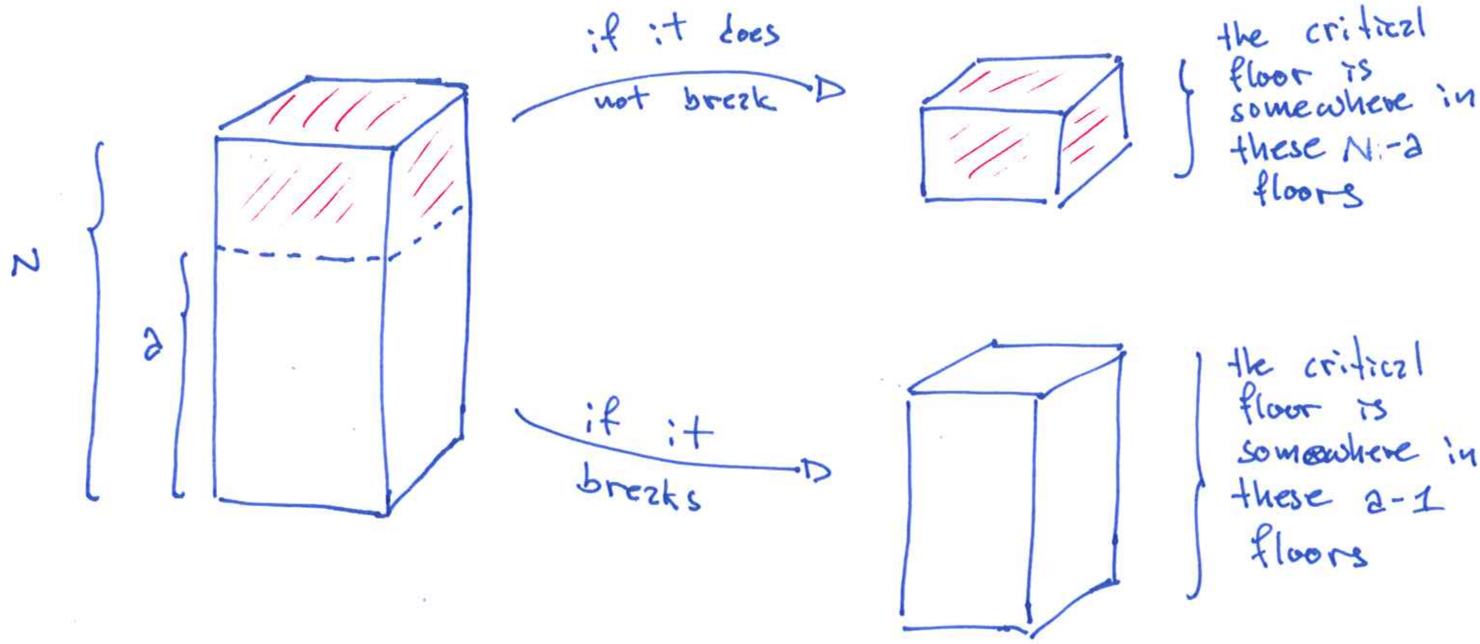
(**Remark:** If the system terminates (he goes to prison), he will not collect any future rewards — it is "game over".)

Ex 2.5

You are in a 100-stories building and have 2 eggs. You want to determine, with the minimum number of drops in the worst case, from which floor it is safe to drop eggs.

Solution:

Notice the modular structure of the problem. If we drop an egg from floor a:



After each drop, we end up in the same situation as before, but with less floors.

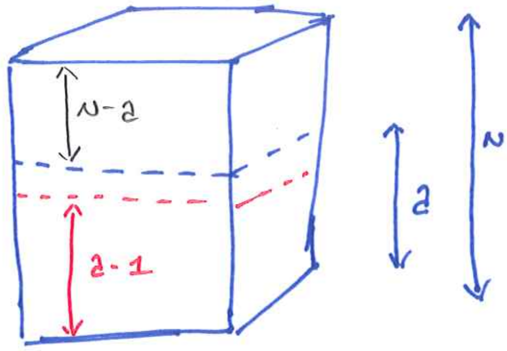
This is typical of problems that can be solved using dynamic programming: the problem can be broken down into a collection of simpler subproblems. Each subproblem is solved only once and its solution stored.

The next time the same subproblem appears, instead of having to recompute its solution, we simply look it up!

Let $V(N, e)$ be the minimum number of drops needed if we have e eggs and N floors to test.

Then

$$V(N,e) = \min_{a=1,\dots,N} \left\{ 1 + \max\left\{ V(N-a,e), V(a-1,e-1) \right\} \right\}$$



if the egg does not break, we will use the minimum # of drops for the upper N-a floors

if it breaks, we will require the minimum # of drops to check the lower a-1 floors with one egg less.

Since we consider the worst-case scenario, we take the maximum of these two.

"Nature is allowed to change floor and picks the one that always will require us to use as many drops as possible"

Performing this test requires one drop, hence the +1.

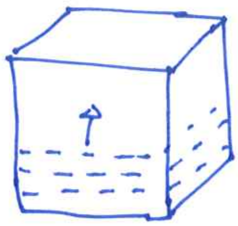
Finally, we try to be optimal, so we minimize with respect to what floor a we drop at.

For the recursion to be well-defined, we need to define the base cases:

- $V(1,e) = 1$

if we are uncertain about one floor, we need to perform exactly one drop to be able to say if it is critical or not

- $V(N,1) = N$



if we have only one egg, to be absolutely certain about which floor is critical, we need to start from the bottom and try all. The worst case is that the last is the critical one.

Evaluating the recursion with these base cases yields that

$$V(100, 2) = 14 \text{ drops}.$$

```python
### How NOT to implement it (solutions of subproblems are recomputed)

import numpy as np

N = 100
E = 2

def V(n, e):
    # Base cases
    if e == 1:
        return n

    if n == 1:
        return 1

    # The DP equation
    minimum = n+1;
    for a in range(1, n+1):
        minimum = min(minimum, max(V(n-a, e), V(a-1, e-1)) + 1)

    return minimum

print "Number of drops needed for %i floors and %i initial eggs:" % (N, E)
print V(N, E)



### How to implement it using dynamic programming (solutions of subproblems are
#   stored and looked up).

import numpy as np

N = 100
E = 2

V_lookup = -np.ones((N, E))

def V(n, e):
    # Base cases
    if e == 1:
        return n

    if n == 1:
        return 1

    # Check if this value has already been computed
    if not V_lookup[n-1, e-1] == -1:
        return V_lookup[n-1, e-1]

    # The DP equation
    minimum = n+1;                # Set to something high
    for a in range(1, n+1):
        minimum = min(minimum, max(V(n-a, e), V(a-1, e-1)) + 1)

    # Save the value
    V_lookup[n-1, e-1] = minimum

    return minimum

print "Number of drops needed for %i floors and %i initial eggs:" % (N, E)
print V(N, E)


### Output ###

# In [1]: %run egg.py
# Number of drops needed for 100 floors and 2 initial eggs:
#    14.0
```

Ex 2.2 Have to sell apartment within N days. We receive an offer w_t every evening that has to be accepted or rejected the following day. The offers are multiples of 10 000 SEK, i.i.d., positive and upper-bounded. Once we sell, we get a daily interest rate $\rho > 0$ on the money we place in the bank.

Solution:

State-space:

The old offers are not relevant once we receive a new (since we cannot call back an old buyer). Hence, we let the state of the MDP be the current bid.

If $10\,000 \cdot x_{max}$ is the highest possible bid, then

(x, x_{max} are integers)

$$\mathcal{S} = \{10\,000 \cdot x \text{ for } 0 \leq x \leq x_{max}\} \cup \{\text{Sold}\}.$$

To simplify notation, let's agree that we always speak in units of 10k SEK, so that

$$\mathcal{S} = \{0, 1, \ldots, x_{max}\} \cup \{\text{Sold}\}.$$

Actions:

Two choices:

$\mathcal{A}$ - Accept offer

$\mathcal{R}$ - Reject offer

Time-horizon and objective:

Finite-horizon N:

$$\mathbb{E}\left\{\sum_{t=0}^{N-1} r_t(s_t, a_t) + r_N(s_N)\right\}$$

Rewards:

Terminal:

- $r_N(\text{Sold}) = 0$
- $r_N(x) = x$

we are forced to sell at the end

Non-terminal:

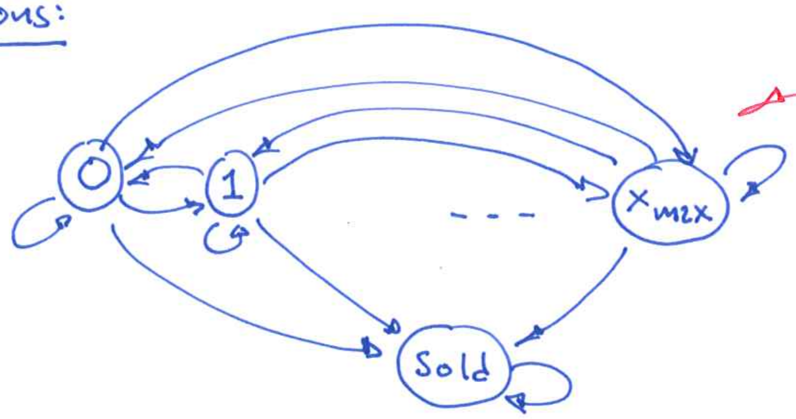
- $r_t(s = x, a = \mathcal{A}) = x \cdot (1+\rho)^{N-t}$

the current offer

future interest

- $r_t(s = \text{Sold}, a = \mathcal{A}) = 0$
- $r_t(s = \text{Sold}, a = R) = 0$
- $r_t(s = x, a = R) = 0$

Transitions:



the next bid can be anything, so all these states are connected

The non-zero transitions are:

- $P_t(s' = x \mid s \neq \text{Sold}, a = R) = \Pr\{\omega_t = x\}$

if we reject, tomorrow's offer is drawn iid from the bids' distribution

- $P_t(s' = \text{Sold} \mid s \neq \text{Sold},\ a = \mathcal{A}) = 1$

we decide to sell

- $P_t(s' = \text{Sold} \mid s = \text{Sold},\ a = \bullet) = 1$

Let's solve the MDP. Recall the backward induction:

- $u_N^*(s_N) = r_N(s_N)$
- $u_t^*(s_t) = \max_a \left\{ r_t(s_t, a) + \sum_{s' \in S} P_t(s' \mid s_t, a)\, u_{t+1}^*(s') \right\}$
- $a_t^*(s_t) \in \arg\max_a \left\{ \text{———} \;''\; \text{———} \right\}$

The base case:

$$u_N^*(\text{Sold}) = r_N(\text{Sold}) = 0$$

$$u_N^*(x) = r_N(x) = x$$

The recursion:

Consider first $s = \text{Sold}$:

$$u_t^*(\text{Sold}) = \max_a \Big\{ \underbrace{r_t(\text{Sold}, a)}_{=0 \text{ for any action}} + \sum_{s'} \underbrace{P_t(s' \mid \text{Sold}, a)}_{=1 \text{ for } s' = \text{Sold}, \text{ zero otherwise}}\, u_{t+1}^*(s') \Big\}$$

$$= \max_a \left\{ 0 + 1 \cdot u_{t+1}^*(\text{Sold}) \right\} =$$

$$= u_{t+1}^*(\text{Sold})$$

By induction, we conclude

$$u_t^*(\text{Sold}) = u_{t+1}^*(\text{Sold}) = \dots = u_N^*(\text{sold}) = 0 \qquad (*)$$

Consider $s = x$:

$$u_t^*(x) = \max_a \left\{ r_t(x,a) + \sum_{s'} p(s' \mid x,a)\, \underbrace{u_{t+1}^*(s')}_{\text{this is zero for } s' = \text{Sold according to } (*)} \right\} =$$

$$= \max_a \left\{ r_t(x,a) + \sum_{y=0}^{x_{max}} p(y \mid x,a)\, u_{t+1}^*(y) \right\} \qquad (**)$$

Let's evaluate the two actions separately:

If $a = A$:

"when we accept, we can only go to sold"

$$\underbrace{r_t(x,A)}_{= x(1+r)^{N-t}} + \sum_{y=0}^{x_{max}} \underbrace{p(y \mid x,A)}_{= 0 \text{ for all } y.} u_{t+1}^*(y) = x(1+r)^{N-t}$$

If $a = R$:

$$\underbrace{r_t(x,R)}_{=0} + \sum_{y=0}^{x_{max}} \underbrace{p(y \mid x,R)}_{= \Pr\{w_t = y\}} u_{t+1}^*(y) =$$

$$\sum_{y=0}^{x_{max}} \Pr\{w_t = y\}\, u_{t+1}^*(y) = E_w\left\{ u_{t+1}^*(w) \right\}$$

Taken together into (**), we have that:

$$u_t^*(x) = \max \Big\{ \underbrace{x(1+r)^{N-t}}_{\text{accept}}, \underbrace{\mathbb{E}_\omega \{ u_{t+1}^*(\omega) \}}_{\text{reject}} \Big\}$$

$$= \max \Big\{ \underbrace{x}_{\text{accept}}, \underbrace{\frac{\mathbb{E}_\omega \{ u_{t+1}^*(\omega) \}}{(1+r)^{N-t}}}_{\text{reject}} \Big\}$$

$$\overset{\text{def.}}{=} \max \{ \underbrace{x}_{A}, \underbrace{\alpha_t}_{R} \}.$$

$\Longrightarrow$ (If we have sold, then the action we take is irrelevant.)

If at time t we receive offer x, then we should:

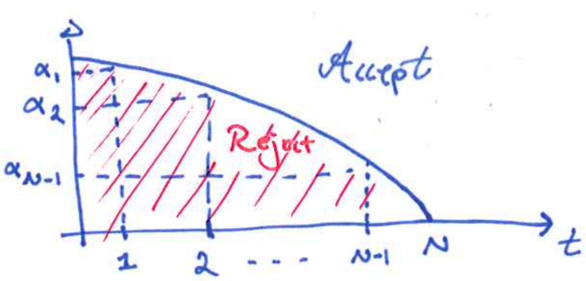
accept the offer if $x > \alpha_t$

reject the offer if $x \leq \alpha_t$

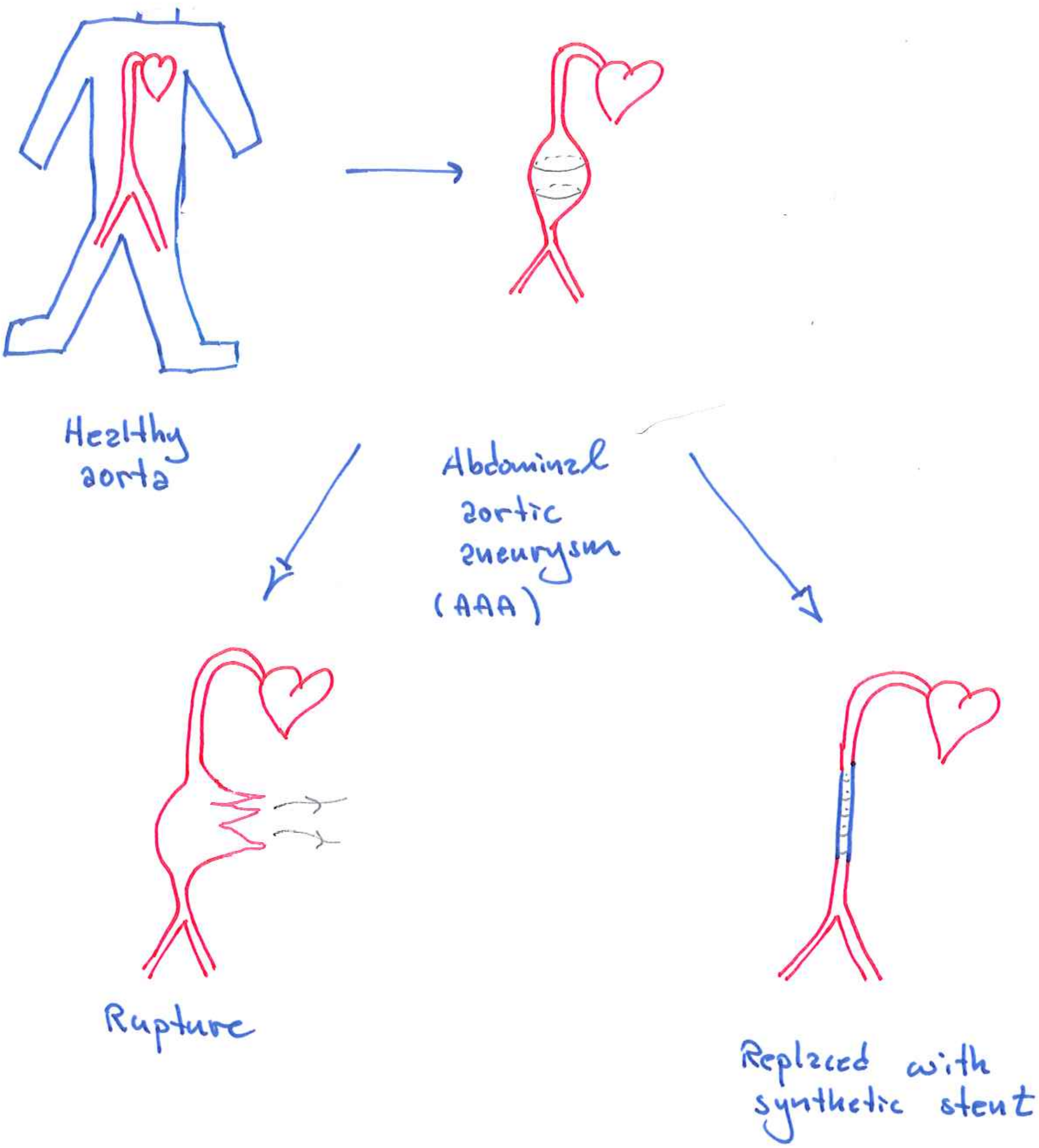
if $x=\alpha_t$, we can do whatever

for $\alpha_t = \dfrac{\mathbb{E}_\omega \{ u_{t+1}^*(\omega) \}}{(1+r)^{N-t}}$.

This is a time-dependent threshold policy:



Ex 2.10



Each year, the AAA can grow or rupture. Risk of rupture is related to its size. A doctor can treat AAA, but risks in surgery grow with patient age.

Model as MDP, with aim to maximize life-expectancy.

Solution:

State-space:

The central quantity is the AAA's size.
To simplify, discretize:

$$\mathcal{S} = \{ 3\text{-}4 \text{ cm}, 4\text{-}5 \text{ cm}, \ldots, 9\text{-}10 \text{ cm}, \text{dead}, \text{healthy} \}$$

AAA diameter

the patient is deceased

state after successful surgery

Actions:

The doctor has two choices every year:

$\mathcal{O}$ - operate (perform surgery),

$\mathcal{N}$ - nothing (continue surveillance)

Rewards:

We aim to maximize life-expectancy, let's give reward 1 for every year patient is alive:

- $r_t(s = \text{dead}, a = \cdot) = 0$,
- all other 1.

Time-horizon and objective:

We identify time t as the patient's age. Assume maximum age of humans is 120 years.

Then

$$\mathbb{E}\left\{ \sum_{t=65}^{T-1} r_t(s_t, a_t) + r_T(s_T) \right\}$$

with $T = 120$.

Remark:

AAAs are not common in patients younger than ca 65 years. The lower index in the sum reflects that we are only interested in a policy in the age range where the disease is prevalent.

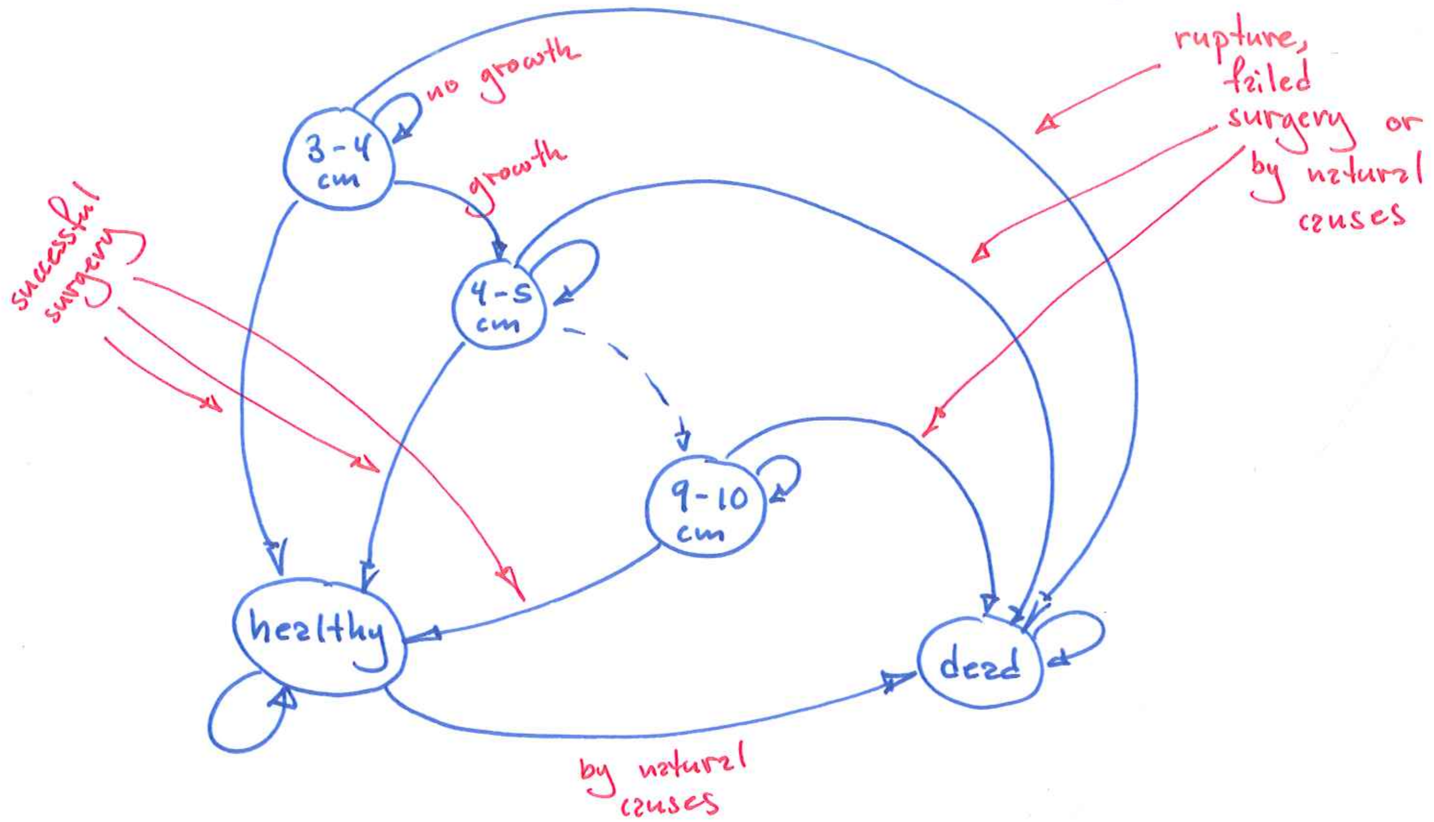
Transition probabilities:

Introduce the following parameters:

- d_t : probability of death by natural causes at age t
- $f_t(\text{size})$: probability of failure in surgery (for AAA of size "size").
- $R(\text{size})$: probability of rupture
- $g(\text{size})$: probability of growing one size

(Modeling assumption, Reasonable due to coarse discretization.)

Let's draw to identify the non-zero transitions:



The non-zero transitions are:

- $P_t(s' = size \mid s = size,\ a = \mathcal{W}) = \underbrace{(1-d_t)}_{\text{no death by natural causes}} \times \underbrace{(1-g(size))}_{\text{no growth}} \times \underbrace{(1-R(size))}_{\text{no rupture}}$

  stay the same

- $P_t(s' = \text{next } size \mid s = size,\ a = \mathcal{W}) = (1-d_t) \times g(size) \times (1-R(size))$

  we grow one size

- $P_t(s' = dead \mid s = size,\ a = \mathcal{W}) = Pr\{\text{rupture or natural death} \mid s = size,\ a = \mathcal{W}\} =$

  $\left\{ \begin{array}{l} \text{Recall:} \\ P(A \cup B) = P(A) + P(B) - P(A \cap B) \\ \quad \text{"or"} \qquad\qquad\qquad\qquad \text{"and"} \end{array} \right\} =$

  $R(size) + d_t - R(size) \times d_t$.

- $P_t(s' = healthy \mid s = size,\ a = \mathcal{O}) = 1 - f_t(size)$ successful surgery
- $P_t(s' = dead \mid s = size,\ a = \mathcal{O}) = f_t(size)$ unsuccessful surgery

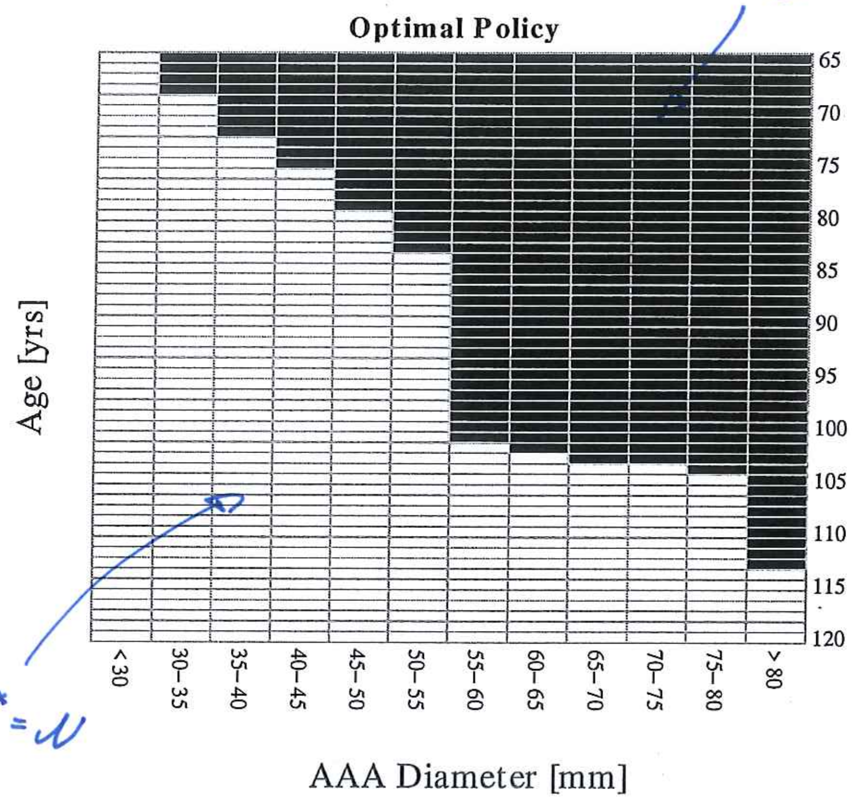
- $P_t(s' = healthy \mid s = healthy,\ a = \bullet) = 1 - d_t$
- $P_t(s' = dead \mid s = healthy,\ a = \bullet) = d_t$

- $P_t(s' = dead \mid s = dead,\ a = \bullet) = 1$

Extra:

with estimated model parameters, you obtain the following optimal policy:

$a^* = O$



$a^* = N$

A black cell indicates surgery, and a white that no action should be taken.

Note that this is a time-dependent threshold policy. We'll see more of these in the next exercise session.