THE UNIVERSITY OF QUEENSLAND
AUSTRALIA
CREATE CHANGE

COMP3702/COMP7702
Artificial Intelligence

Module 3: Reasoning and planning under uncertainty — Part 2

Dr Archie Chapman

Semester 2, 2020

The University of Queensland
School of Information Technology and Electrical Engineering

Thanks to Dr Alina Bialkowski and Dr Hanna Kurniawati

Week 8: Logistics

- **Assignment 2 EXTENDED! — due Monday Sept 28**
- If you haven't seen them, Tutorials 5 and 6 will help you with Assignment 2
- Marking Assignment 1 is almost complete
- RiPPLE round 3 has opened — please create resources only for Module 3 in Round 3.
- Assignment 3 will be released during the mid-semester break.
- Next week is the mid-semester break

Week 8 — Overview

Continue Module 3: Reasoning and planning under uncertainty

- Decision theory, continued
- Markov decision processes 1: Representation and exact algorithms

Table of contents

Decision theory

Preferences

- Actions result in outcomes
- Agents have **preferences** over outcomes
- A rational agent will do the action that has the best outcome for them
- Sometimes agents don't know the outcomes of the actions, but they still need to compare actions
- Agents have to act.
  (Doing nothing is (often) an action).

Preferences Over Outcomes

Some notation:

- The preference relation, $\succ$, means "is preferred to" or "succeeds in a preference order"
- $\prec$ is "precedes in a preference order"
- Indifference is $\sim$

If o_1 and o_2 are outcomes

- $o_1 \succeq o_2$ means o_1 is at least as desirable as o_2.
- $o_1 \sim o_2$ means $o_1 \succeq o_2$ and $o_2 \succeq o_1$.
- $o_1 \succ o_2$ means $o_1 \succeq o_2$ and $o_2 \not\succeq o_1$

Lotteries

- An agent may not know the outcomes of its actions, but only have a probability distribution of the outcomes.
- o_1 lottery is a probability distribution over outcomes.
- R&N denote this $[p_1, o_1; p_2, o_2, \ldots, p_k, k]$, and
  P&M denote it: $[p_1 : o_1, p_2 : o_2, \ldots, p_k : k]$ (like a python dict)
  where the i are outcomes and $p_i \geq 0$ such that $\sum_i p_i = 1$
- The lottery specifies that outcome i occurs with probability p_i.
- When we talk about outcomes, we will include lotteries over "pure" outcomes.

Axioms of rational preferences

Idea: preferences of a rational agent must obey certain rules.

Rational preferences imply behaviour describable as maximization of expected utility

Completeness (for some reason R&N call this *Orderability*): $(o_1 \succ o_2) \vee (o_2 \prec o_1) \vee (o_1 \sim o_2)$

Transitivity: $(o_1 \succ o_2) \wedge (o_2 \succ C) \Rightarrow (o_1 \succ C)$

Monotonicity: $o_1 \succ o_2 \Rightarrow (p \geq q \Leftrightarrow [p : o_1, 1-p : o_2] \succeq [q : o_1, 1-q : o_2])$

Continuity: $o_1 \succ o_2 \succ C \Rightarrow \exists p \in [0,1][p : o_1, 1-p : C] \sim o_2$

Substitutability: $o_1 \sim o_2 \Rightarrow [p : o_1, 1-p : C] \sim [p : o_2, 1-p : C]$

Decomposability $[p : o_1, 1-p : [q : o_2, 1-q : o_3]] \sim [p : o_1, (1-p)q : o_2, (1-p)(1-q)o_3]$

Properties of Preferences — Completeness

Completeness: Agents have to act, so they must have preferences:

$$\forall o_1 \forall o_2 \; o_1 \succeq o_2 \text{ or } o_2 \succeq o_1$$

Properties of Preferences — Transitivity

Transitivity: Preferences must be transitive:

$$\text{if } o_1 \succeq o_2 \text{ and } o_2 \succ o_3 \text{ then } o_1 \succ o_3$$

(Similarly for other mixtures of $\succ$ and $\succeq$.)

Rationale: otherwise $o_1 \succeq o_2$ and $o_2 \succ o_3$ and $o_3 \succeq o_1$.
If they are prepared to pay to get o_2 instead of o_3,
and are happy to have o_1 instead of o_2,
and are happy to have o_3 instead of o_1
$\longrightarrow$ money pump.

Properties of Preferences — Monotonicity

Monotonicity: An agent prefers a larger chance of getting a better outcome than a smaller chance:

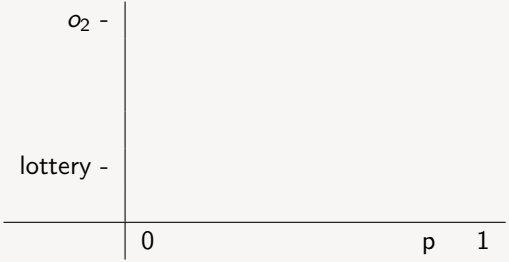
- If $o_1 \succ o_2$ and $p > q$ then

$$[p : o_1, 1 - p : o_2] \succ [q : o_1, 1 - q : o_2]$$

Consequence of axioms of Completeness, Transitivity and Monotonicity

- Suppose $o_1 \succ o_2$ and $o_2 \succ o_3$. Consider whether the agent would prefer
  - o_2
  - the lottery $[p : o_1, 1 - p : o_3]$

  for different values of $p \in [0, 1]$.
- Plot which one is preferred as a function of p:



Properties of Preferences — Continuity

Continuity: Suppose $o_1 \succ o_2$ and $o_2 \succ o_3$, then there exists a $p \in [0, 1]$ such that

$$o_2 \sim [p : o_1, 1 - p : o_3]$$

Properties of Preferences — Substitutability

Substitutability: if $o_1 \sim o_2$ then the agent is indifferent between lotteries that only differ by o_1 and o_2:

$$[p : o_1, 1 - p : o_3] \sim [p : o_2, 1 - p : o_3]$$

Alternative Axiom for Substitutability

Substitutability: if $o_1 \succeq o_2$ then the agent weakly prefers lotteries that contain o_1 instead of o_2, everything else being equal.

That is, for any number p and outcome o_3:

$$[p : o_1, (1-p) : o_3] \succeq [p : o_2, (1-p) : o_3]$$

Properties of Preferences — Decomposability

Decomposability: (no fun in gambling). An agent is indifferent between lotteries that have same probabilities and outcomes. This includes lotteries over lotteries.

For example:

$$\begin{aligned}&[p : o_1, 1-p : [q : o_2, 1-q : o_3]]\\ \sim\quad &[p : o_1, (1-p)q : o_2, (1-p)(1-q) : o_3]\end{aligned}$$

Summary of Rational Preferences

Completeness:

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

Transitivity:

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \Rightarrow (o_1 \succ C)$$

Monotonicity:

$$o_1 \succ o_2 \Rightarrow (p \geq q \Leftrightarrow [p : o_1, 1-p : o_2] \succeq [q : o_1, 1-q : o_2])$$

Continuity:

$$o_1 \succ o_2 \succ C \Rightarrow \exists p \in [0,1][p : o_1, 1-p : C] \sim o_2$$

Substitutability:

$$o_1 \sim o_2 \Rightarrow [p : o_1, 1-p : C] \sim [p : o_2, 1-p : C]$$

Decomposability $[p : o_1, 1-p : [q : o_2, 1-q : o_3]] \sim [p : o_1, (1-p)q : o_2, (1-p)(1-q)o_3]$

What we would like

- We would like a measure of preference that can be combined with probabilities. So that

$$\begin{aligned} & \mathit{value}([p : o_1, 1-p : o_2]) \\ = \quad & p \times \mathit{value}(o_1) + (1-p) \times \mathit{value}(o_2) \end{aligned}$$

- Money does not act like this.
  What would you prefer

$$\$1,000,000 \text{ or } [0.5 : \$0, 0.5 : \$2,000,000]?$$

- It may seem that preferences are too complex and muti-faceted to be represented by single numbers.

Theorem

If preferences follow the preceding properties, then preferences can be measured by a function

$$utility : outcomes \rightarrow [0, 1]$$

such that

- $o_1 \succ o_2$ if and only if $utility(o_1) \geq utility(o2)$
- $o_1 \sim o_2$ if and only if $utility(o_1) = utility(o2)$
- Utilities are linear with probabilities:

$$\begin{aligned} & utility([p_1 : o_1, p_2 : o_2, \ldots, p_k : o_k]) \\ = & \sum_{i=1}^{k} p_i \times utility(o_i) \end{aligned}$$

So, we can replace preferences with real-numbers, but still what is the "best" lottery?

Maximum Expected Utility (MEU)

Main problem: What does "best" mean?

Utility: a number that assigns the desirability of a state, MEU is the commonly used definition of "best" decision

Idea:

- Assigns utility function to each outcome (state) to represent the agent's preference
- "Best" decision maximises the expected utility of the outcomes

Example: Buying a Used Car

Goal of buying the car: To gain profit from reselling it

Car costs $1000

Can sell the car for $1100 $\Rightarrow$ $100 profit

BUT Every car is either good or bad

- Costs $40 to repair a good car
- Costs $200 to repair a bad car
- 20% cars are bad

Should we buy the car? $\rightarrow$ Solve using MEU

Example: Buying a Used Car

State space: {good car, bad car}

Preference: good car $\succ$ bad car

Utility function:

- U(good car) =1100 - 1000 - 40 = 60
- U(bad car) = 1100 - 1000 - 200 = -100

Lottery: [0.8, good car; 0.2, bad car]

Expected Utility if we buy the car:

- $P(goodcar) \times U(goodcar) + P(badcar) \times U(badcar) = 0.8 \times 60 + 0.2 \times -100 = 28$
- Higher than not buying the car. Hence, **buy!**

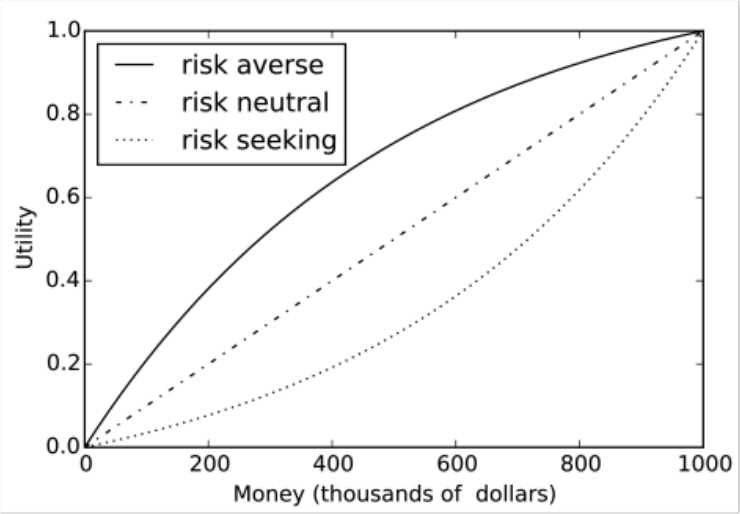
How about the Utility of Money?

Which one do you prefer?
A: A sure gain of \$240 **B**: A 25% chance of winning \$1000 and 75% chance of winning nothing

Which one do you prefer? **C**: A sure loss of \$750 **D**: A 75% chance of losing \$1000 and 25% chance of losing nothing

Is decision theory useless here? Need a better utility function that can incorporate our preference! (must follow the axioms)

Utility as a function of money



Factored Representation of Utility

- Suppose the outcomes can be described in terms of features $X_1, \ldots, X_n$.
- An additive utility is one that can be decomposed into set of factors:

$$u(X_1, \ldots, X_n) = f_1(X_1) + \cdots + f_n(X_n).$$

  This assumes additive independence.
- Strong assumption: contribution of each feature doesn't depend on other features.
- Many ways to represent the same utility:
  — a number can be added to one factor as long as it is subtracted from others.

Additive Utility

- An additive utility has a canonical representation:

$$u(X_1, \ldots, X_n) = w_1 \times u_1(X_1) + \cdots + w_n \times u_n(X_n).$$

- If $best_i$ is the best value of X_i, $u_i(X_i{=}best_i) = 1$.
  If $worst_i$ is the worst value of X_i, $u_i(X_i{=}worst_i) = 0$.
- w_i are weights, $\sum_i w_i = 1$.
  The weights reflect the relative importance of features.
- We can determine weights by comparing outcomes.

$$w_1 = u(best_1, x_2, \ldots, x_n) - u(worst_1, x_2, \ldots, x_n).$$

  for any values $x_2, \ldots, x_n$ of $X_2, \ldots, X_n$.

Complements and Substitutes

- Often additive independence is not a good assumption.
- Values x_1 of feature X_1 and x_2 of feature X_2 are complements if having both is better than the sum of the two.
- Values x_1 of feature X_1 and x_2 of feature X_2 are substitutes if having both is worse than the sum of the two.
- Example: on a holiday
  - An excursion for 6 hours North on day 3.
  - An excursion for 6 hours South on day 3.
- Example: on a holiday
  - A trip to a location 3 hours North on day 3
  - The return trip for the same day.

Generalized Additive Utility

- A generalized additive utility can be written as a sum of factors:

$$u(X_1, \ldots, X_n) = f_1(\overline{X_1}) + \cdots + f_k(\overline{X_k})$$

  where $\overline{X_i} \subseteq \{X_1, \ldots, X_n\}$.
- An intuitive canonical representation is difficult to find.
- It can represent complements and substitutes.

Utility and time

- Would you prefer $1000 today or $1000 next year?
- What price would you pay now to have an eternity of happiness?
- How can you trade off pleasures today with pleasures in the future?

Utility and time

- How would you compare the following sequences of rewards (per week):

  A: \$1000000, \$0, \$0, \$0, \$0, \$0,...

  B: \$1000, \$1000, \$1000, \$1000, \$1000,...

  C: \$1000, \$0, \$0, \$0, \$0,...

  D: \$1, \$1, \$1, \$1, \$1,...

  E: \$1, \$2, \$3, \$4, \$5,...

Rewards and Values

Suppose the agent receives a sequence of rewards $r_1, r_2, r_3, r_4, \ldots$ in time. What utility should be assigned? "Return" or "value"

- total reward $V = \sum_{i=1}^{\infty} r_i$
- average reward $V = \lim_{n \to \infty} (r_1 + \cdots + r_n)/n$

Rewards and Values

Suppose the agent receives a sequence of rewards $r_1, r_2, r_3, r_4, \ldots$ in time.

- discounted return $V = r_1 + \gamma r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \cdots$
  γ is the discount factor $0 \leq \gamma \leq 1$.

Properties of the Discounted Rewards

- The discounted return for rewards $r_1, r_2, r_3, r_4, \ldots$ is

$$\begin{aligned} V &= r_1 + \gamma r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \cdots \\ &= r_1 + \gamma(r_2 + \gamma(r_3 + \gamma(r_4 + \ldots))) \end{aligned}$$

- If V_t is the value obtained from time step t

$$V_t = r_t + \gamma V_{t+1}$$

- How is the infinite future valued compared to immediate rewards?
  $1 + \gamma + \gamma^2 + \gamma^3 + \cdots = 1/(1-\gamma)$
  Therefore $\dfrac{\text{minimum reward}}{1-\gamma} \leq V_t \leq \dfrac{\text{maximum reward}}{1-\gamma}$
- We can approximate V with the first k terms, with error:

$$V - (r_1 + \gamma r_2 + \cdots + \gamma^{k-1} r_k) = \gamma^k V_{k+1}$$

Allais Paradox (1953)

What would you prefer:

A: $\$1m$ — one million dollars

B: lottery $[0.10 : \$2.5m, 0.89 : \$1m, 0.01 : \$0]$

What would you prefer:

C: lottery $[0.11 : \$1m, 0.89 : \$0]$

D: lottery $[0.10 : \$2.5m, 0.9 : \$0]$

It is inconsistent with the axioms of preferences to have $A \succ B$ and $D \succ C$.

A,C: lottery $[0.11 : \$1m, 0.89 : X]$

B,D: lottery $[0.10 : \$2.5m, 0.01 : \$0, 0.89 : X]$

Framing Effects [Tversky and Kahneman]

- A disease is expected to kill 600 people. Two alternative programs have been proposed:

  Program A: 200 people will be saved

  Program B: probability 1/3: 600 people will be saved
  probability 2/3: no one will be saved

  Which program would you favor?

- A disease is expected to kill 600 people. Two alternative programs have been proposed:

  Program C: 400 people will die

  Program D: probability 1/3: no one will die
  probability 2/3: 600 will die

  Which program would you favor?

Tversky and Kahneman: 72% chose A over B.

22% chose C over D.

Decision-theoretic planning

Agents as Processes

Agents carry out actions:

- forever infinite horizon
- until some stopping criteria is met indefinite horizon
- finite and fixed number of steps finite horizon

Decision-theoretic Planning

What should an agent do when

- it gets rewards (and penalties) and tries to maximize its rewards received
- actions can be stochastic; the outcome of an action can't be fully predicted
- there is a model that specifies the (probabilistic) outcome of actions and the rewards
- the world is fully observable?

Initial Assumptions for decision-theoretic planning

- flat or modular or hierarchical
- explicit states or features or individuals and relations
- static or finite stage or indefinite stage or infinite stage
- fully observable or partially observable
- deterministic or stochastic dynamics
- goals or complex preferences
- single agent or multiple agents
- knowledge is given or knowledge is learned
- perfect rationality or bounded rationality

Markov decision processes

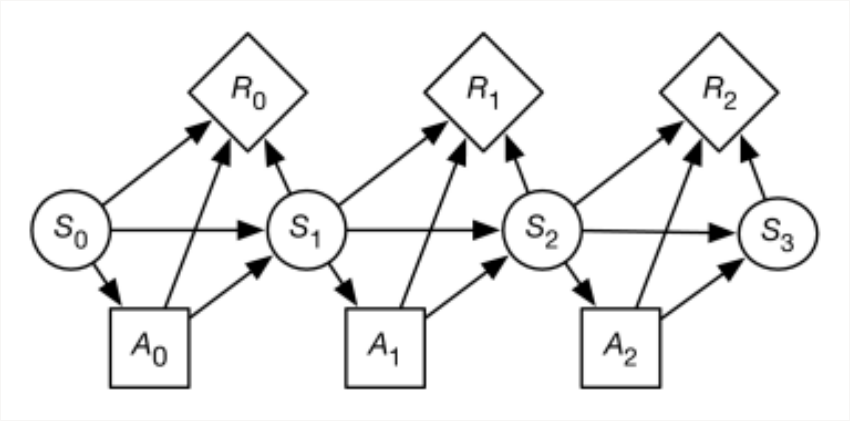
World State

- The world state is the information such that if the agent knew the world state, no information about the past is relevant to the future. Markovian assumption.
- S_k is state at time k, and A_k is the action at time k:

$$P(S_{t+1} \mid S_0, A_0, \ldots, S_t, A_t) = P(S_{t+1} \mid S_t, A_t)$$

  $P(s' \mid s, a)$ is the probability that the agent will be in state s' immediately after doing action a in state s.
- The dynamics is stationary if the distribution is the same for each time point.

- A Markov decision process augments a Markov chain with actions and values:



Markov Decision Processes

An MDP consists of:

- set S of **states**.
- set A of **actions**.
- $P(S_{t+1} \mid S_t, A_t)$ specifies the dynamics or **transition function**.
- $R(S_t, A_t, S_{t+1})$ specifies the **reward** at time t.
  Sometimes is a random variable, $R(s, a, s')$ is the expected reward received when the agent is in state s, does action a and ends up in state s'.
  Sometimes we use $R(s, a) = \sum_{s'} P(s' \mid s, a) R(s, a, s')$.
- γ is **discount factor**.
- An MDP's **objective** is: $\mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right]$.

Example: to exercise or not?

Each week *Archie* has to decide whether to exercise or not:

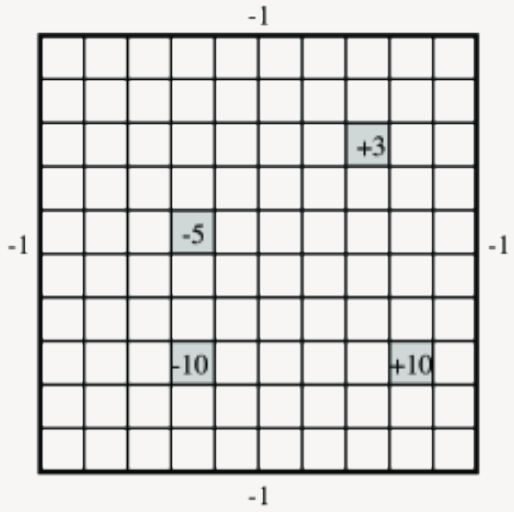
- States: $\{fit, unfit\}$
- Actions: $\{exercise, relax\}$
- Dynamics:

State	Action	$P(fit \mid State, Action)$
fit	exercise	0.99
fit	relax	0.7
unfit	exercise	0.2
unfit	relax	0.0

- Reward (does not depend on resulting state):

State	Action	Reward
fit	exercise	8
fit	relax	10
unfit	exercise	0
unfit	relax	5

Example: Simple Grid World



Grid World Model

- **Actions**: up, down, left, right.
- **States**: 100 states corresponding to the positions of the robot.
- **Transitions**: Robot goes in the commanded direction with probability 0.7, and one of the other directions with probability 0.1.
- **Rewards**: If it crashes into an outside wall, it remains in its current position and has a reward of -1.
- Four special rewarding states; the agent gets the reward when leaving.

Planning Horizons

The planning horizon is how far ahead the planner looks to make a decision.

- The robot gets flung to one of the corners at random after leaving a positive ($+10$ or $+3$) reward state.
  - the process never halts
  - infinite horizon
- The robot gets $+10$ or $+3$ in the state, then it stays there getting no reward. Or it is left with only a special action **exit**, and the *episode* ends. These are absorbing states.
  - the robot will eventually reach an absorbing state. WHY?
  - indefinite horizon

Information Availability

What information is available when the agent decides what to do?

- fully-observable MDP the agent gets to observe S_t when deciding on action A_t.
- partially-observable MDP (POMDP) the agent has some noisy sensor of the state. It is a mix of a hidden Markov model and MDP. It needs to remember (some function of) its sensing and acting history.

[This lecture only considers FOMDPs]

Policies

- A policy is a sequence of actions, taken to move from each state to the next state over the whole time horizon.
- A stationary policy is a function or a map:

$$\pi : S \to A$$

  Given a state s, $\pi(s)$ specifies what action the agent who is following π will do.
- An optimal policy is one with maximum expected discounted reward.

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0} \gamma^t R(s_t, \pi(s_t))\right]$$

  where $\pi(t)$ is the action taken at time t.
- For a fully-observable MDP with stationary dynamics and rewards with infinite or indefinite horizon, there is always an optimal stationary policy.

Example: to exercise or not?

Each week *Archie* has to decide whether to exercise or not:

- States: $\{fit, unfit\}$
- Actions: $\{exercise, relax\}$

How many stationary policies are there?

What are they?

For the grid world with 100 states and 4 actions
How many stationary policies are there?

Value of a Policy

We first approach MDPs using a recursive reformulation of the objective called a value function

- The value function of an MDP, $V^{\pi}(s)$, is the expected future cost of following an (arbitrary) policy, π, starting from state, s, given by:

$$V^{\pi}(s) = \sum_{s' \in \mathcal{S}} P(s' \mid \pi(s), s) \left[R(s, \pi(s), s') + \gamma V^{\pi}(s') \right].$$

  where the policy $\pi(s)$ determines that action taken in state s.

- Here we have dropped the time index, as it is redundant, but note that $a_t = \pi(s_t)$.

Value of a Policy

Given a policy π:

- The Q-function represents the value of choosing an action and them following policy π in every subsequent state.
- $Q^\pi(s, a)$, where a is an action and s is a state, is the expected value of doing a in state s, then following policy π.
- Q^π and V^π can be defined mutually recursively:

$$\begin{aligned} Q^\pi(s,a) &= \sum_{s'} P(s' \mid a, s)\left(R(s,a,s') + \gamma V^\pi(s')\right) \\ V^\pi(s) &= Q(s, \pi(s)) \end{aligned}$$

Computing the Value of a Policy

Let $v^\pi \in \mathbb{R}^{|S|}$ be a vector of values for each state, and $r \in \mathbb{R}^{|S|}$ be a vector of rewards for each state.

Let $P^\pi \in R^{|S| \times |S|}$ be a matrix containing probabilities for each transition under policy π, where:

$$P^\pi_{ij} = P(s_{t+1} = j \mid s_t = i, a_t = \pi(s_t))$$

Then the value function can be written in vector form as:

$$v^\pi = r + \gamma P^\pi v^\pi$$

We can solve this using linear algebra:

$$\Rightarrow (I - \gamma P^\pi) v^\pi = r$$
$$\Rightarrow v^\pi = (I - \gamma P^\pi)^{-1} r$$

i.e., computing value for a policy requires solving a linear system.

Value of the Optimal Policy

- An **optimal policy**, π^*, expressed in terms of the value function, is one that satisfies *Bellman's optimality condition (1957)*:

$$V^*(s) = \max_a \left\{ R(s, a, s') + \gamma \max_\pi V^\pi(s') \right\}$$

- $V^*(s)$, where s is a state, is the expected value of following the optimal policy in state s.
- Similarly, $Q^*(s, a)$, where a is an action and s is a state, is the expected value of doing a in state s, then following the optimal policy.
- Q^* and V^* can be defined mutually recursively:

$$\begin{aligned} Q^*(s, a) &= \sum_{s'} P(s' \mid a, s) \left(R(s, a, s') + \gamma V^*(s') \right) \\ V^*(s) &= \max_a Q(s, a) \\ \pi^*(s) &= \arg\max_a Q(s, a) \end{aligned}$$

Value Iteration

Value Iteration

- Let V_k be k-step lookahead value function.
- Idea: Given an estimate of the k-step lookahead value function, determine the $k+1$ step lookahead value function.

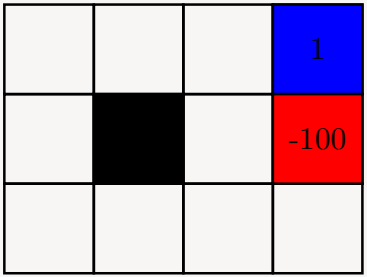
1. Set V_0 arbitrarily, e.g.:
$$\hat{V}(s) \leftarrow 0$$
2. Compute V_{i+1} from V_i.
$$\hat{V}(s) = \max_a \left\{ R(s, a, s') + \hat{V}(s') \right\}$$

Once the values converge, recover the best policy from the current value function estimate:

$$\arg\max_a \ \mathbb{E}\left[R(s, a, s') + \hat{V}(s') \right]$$

- No guarantee we'll reach optimal in finite time, but this converges exponentially fast (in k) to the optimal value function.
- The error reduces proportionally to $\dfrac{\gamma^k}{1-\gamma}$

Grid world



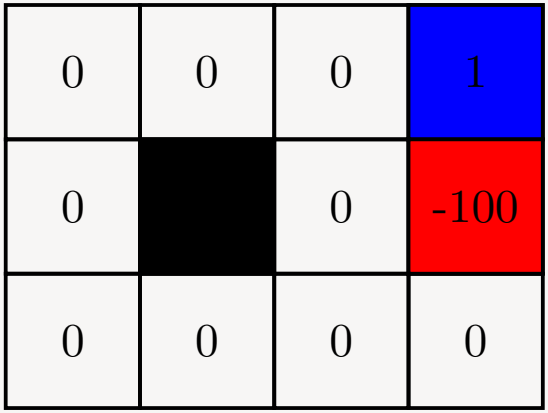
$\gamma = 0.9$

Agent can move up, down, left or right.

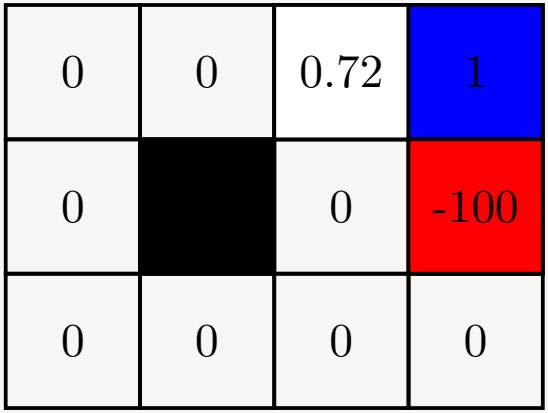
Moves successfully with $p = 0.8$, or perpendicular to direction with $p = 0.1$, each direction.

Hit a wall and the agent stays where it is.

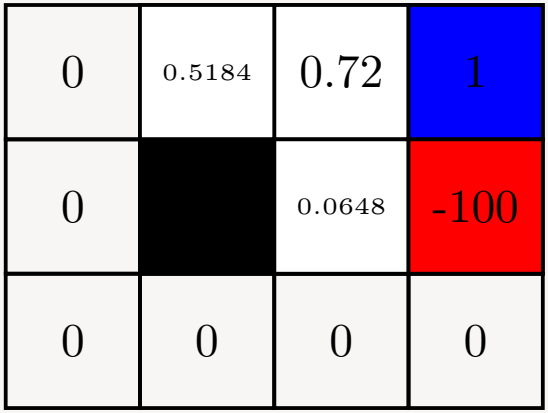
VI — initialisation



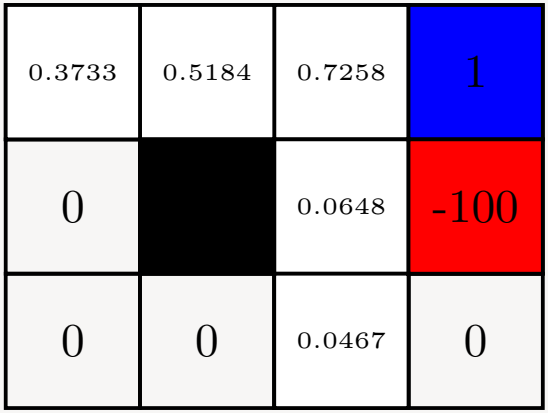
VI — 1 iteration



VI — 2 iterations



VI — 3 iterations



VI — termination

0.6310	0.7282	0.8294	1
0.5540		0.3860	-100
0.4800	0.4215	0.3717	0.1760

Asynchronous Value Iteration

- The agent doesn't need to sweep through all the states, but can update the value functions for each state individually.
- Do update assignments to the states in any order we want, even random
- This converges to the optimal value functions, if each state and action is visited infinitely often in the limit.
- It can either store $V[s]$ or $Q[s, a]$.

Asynchronous VI

Storing $V[s]$

- Repeat forever:
  1. Select state s
  2. $V[s] \leftarrow \max_a \sum_{s'} P(s' \mid s, a) \left(R(s, a, s') + \gamma V[s']\right)$

Storing $Q[s, a]$

- Repeat forever:
  1. Select state s, action a
  2. $Q[s, a] \leftarrow \sum_{s'} P(s' \mid s, a) \left(R(s, a, s') + \gamma \max_{a'} Q[s', a']\right)$

Policy Iteration

Policy Iteration

- Set π_0 arbitrarily, let $i = 0$
- Repeat:
  1. Solve for $V^{\pi_i}(s)$ (or $Q^{\pi_i}(s, a)$):

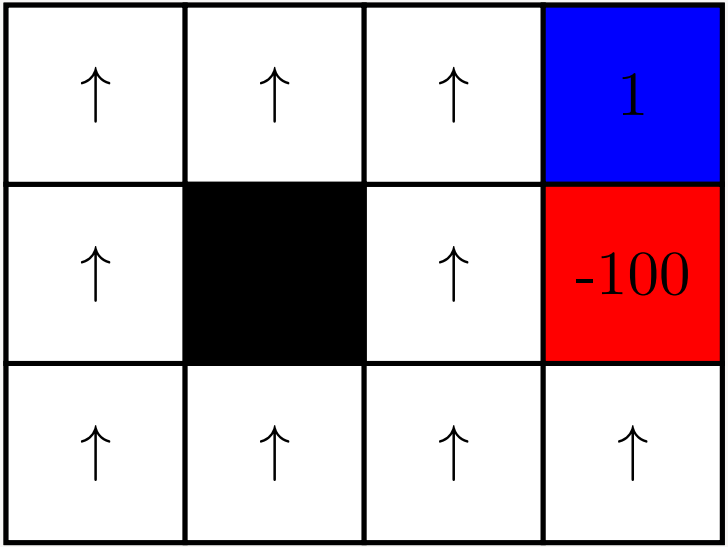
$$V^{\pi_i}(s) = \sum_{s' \in \mathcal{S}} P(s' \mid \pi_i(s), s) \left[R(s, \pi_i(s), s') + \gamma V^{\pi_i}(s') \right] \quad \forall s \in S$$

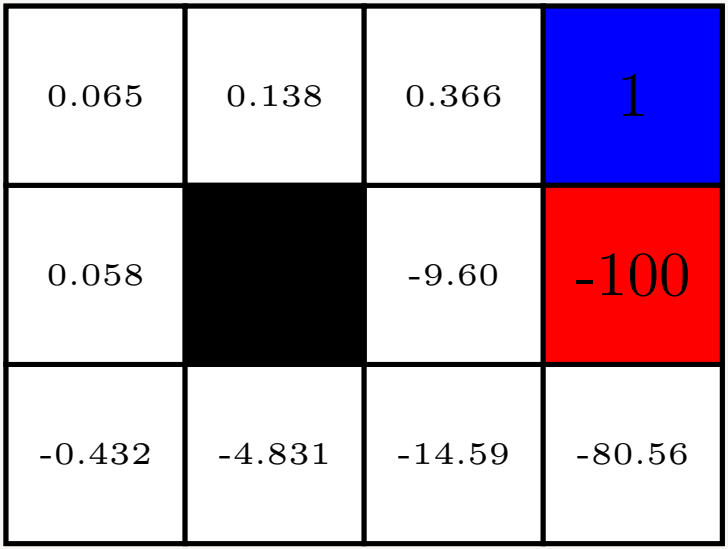
  2. Update policy:

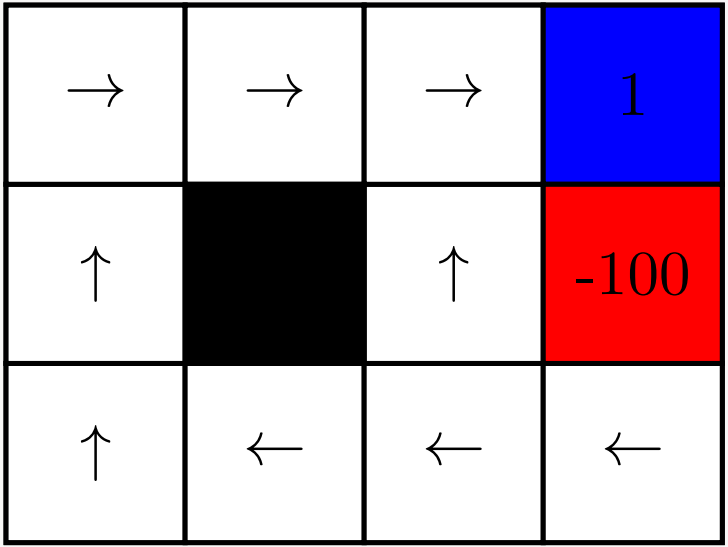
$$\pi_{i+1}(s) \leftarrow \arg\max_a \sum_{s' \in \mathcal{S}} P(s' \mid a, s) \left[R(s, a, s') + \gamma V^{\pi_i}(s') \right]$$

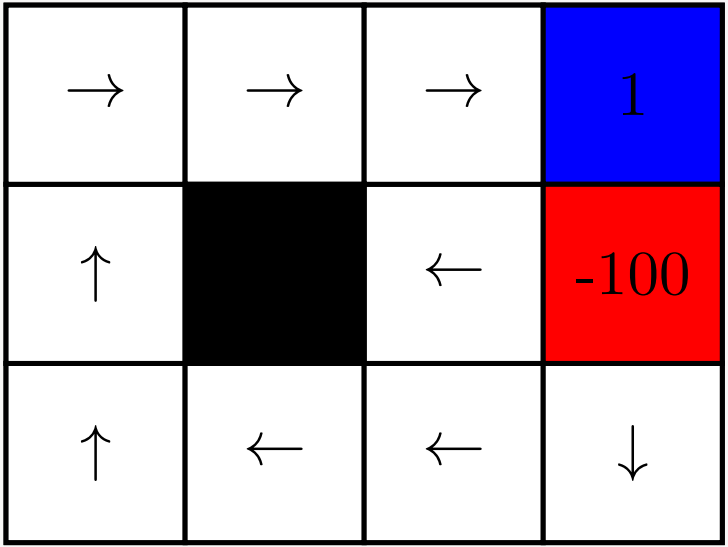
  3. $i = i + 1$
- until $\pi_i(s) = \pi_{i-1}(s)$

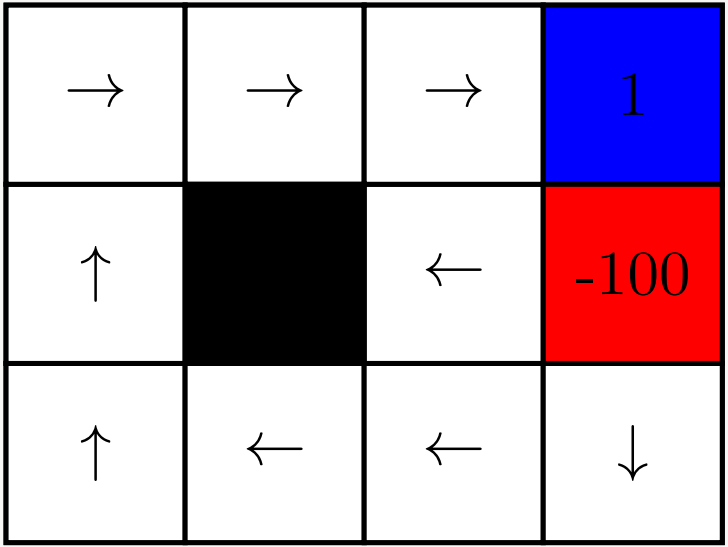
Solving $C^{\pi_i}(s)$ means finding a solution to a set of $|S| \times |A|$ linear equations with $|S| \times |A|$ unknowns, as shown earlier.


1
-100


0.065
0.138
0.366
1
0.058
-9.60
-100
-0.432
-4.831
-14.59
-80.56


1
-100


1
-100


1
-100

Modified Policy Iteration

Set $\pi[s]$ arbitrarily
Set $Q[s, a]$ arbitrarily
Repeat forever:

- Repeat for a while:
  - Select state s, action a
  - $Q[s, a] \leftarrow \sum_{s'} P(s' \mid s, a) \left(R(s, a, s') + \gamma Q[s', \pi[s']]\right)$
- $\pi[s] \leftarrow argmax_a Q[s, a]$

Special case: Finite Horizon MDPs

For finite horizon MDPs, can use dynamic programming or "backwards induction" to compute the optimal value function:

- Start from the goal state and propagate values backwards through the transition function.
- At each decision node, set the value function equal to

$$V^*(s) = \max_a \left\{ R(s, a, s') + \max_\pi V^\pi(s') \right\}$$

- For more detail, see P&M Ch 3.8.3:
  `https://artint.info/2e/html/ArtInt2e.Ch3.S8.SS3.html`

Summary: Q, V, π, R

$$
\begin{aligned}
Q^*(s,a) &= \sum_{s'} P(s' \mid a,s)\left(R(s,a,s') + \gamma V^*(s')\right) \\
V^*(s) &= \max_a Q(s,a) \\
\pi^*(s) &= argmax_a Q(s,a)
\end{aligned}
$$

Let

$$
R(s,a) = \sum_{s'} P(s' \mid a,s) R(s,a,s')
$$

Then:

$$
Q^*(s,a) = R(s,a) + \gamma \sum_{s'} P(s' \mid a,s) V^*(s')
$$

Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS)

Alternative approach: Given a starting state, learn the (local) value model well enough to take an action in the current state, take an action, then estimate the value of the next node.

- Monte Carlo sampling is a well known method for searching through large state space.
- In MDPs, Monte Carlo tree search does this by making use of a generative model, or simulator, of the system under control.
- exploiting MC in sequential decision making was first successfully demonstrated by Kocsis & Szepesvari, 2006 — a relatively recent development.

MCTS can be used as a planning method (offline), or a learning method (online).

E.g. in the online case, we can use simulations to learn the local model, then take an action in the real world and find out what state we actually end up in next.

For now, we use MCTS offline, as a planner, but this will change in Module 4.

Monte Carlo methods

Monte Carlo Simulation: a technique that can be used to solve a mathematical or statistical problem using repeated sampling to determine the properties of some phenomenon (or behavior)

Monte-Carlo Planning: compute a good policy for an MDP by interacting with an MDP simulator

Monte Carlo Tree Search (MCTS)

MCTS is used for sequential decision making, over different states:

- gradually grow the search tree
- two types of tree nodes (and and-or trees)
  1. decision nodes (action selection) — the algorithm selects
  2. chance nodes (world selection) — the world selects the outcome (in the case of MDPs, these are based on known probabilities)
- returned solution: path (action from root) visited the most often

Monte Carlo Tree Search (MCTS)

Gradually grow the search tree:

- Iterate over a tree-walk:
  1. **Bandit phase** Select action from existing tree
  2. **Add a node** Grow a leaf on the fringe of the search tree
  3. **Random phase/roll-out** Select next action to expand from fringe
  4. **Evaluate** Compute instant reward
  5. **Back-propagate** Update information in visited nodes, (like is done in dynamic programming for finite horizon MDPs).
- Returned solution: Path visited most often.

There are lots of details, and we will cover an example in the tutorial next week.

Attributions and References

Thanks to Dr Alina Bialkowski and Dr Hanna Kurniawati for their materials.

Other materials derived from Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, 3E, Prentice Hall, 2009.

All remaining errors are Archie's — please email if you find any: archie.chapman@uq.edu.au