Making (even more) Complex Decisions

Paolo Turrini

Department of Computing, Imperial College London

Introduction to Artificial Intelligence
2nd Part
AlphaGo beats World Go Champion

THE FIRST COMPUTER PROGRAM TO EVER BEAT A PROFESSIONAL PLAYER AT THE GAME OF GO.
Outline

- Rewind
- The Value Iteration Algorithm
The main reference

Stuart Russell and Peter Norvig
Artificial Intelligence: a modern approach
Chapters 17
Begin at the start state

The game ends when we reach either goal state $+1$ or $-1$

Collision results in no movement

Rewards: $+1$ and $-1$ for terminal states respectively, $-0.04$ for all others
The World

- Fully observable
- Markovian
- Discounted rewards
- Stochastic actions
The Agent

Each time it’s like throwing an unfair dice
The Agent

Each time it’s like throwing an unfair dice
The Agent

Each time it’s like throwing an unfair dice
The Agent

Each time it’s like throwing an unfair dice
The Agent

Each time it’s like throwing an unfair dice
Walking is a repetition of throws:

- The probability that I walk right the first time: 0.8
- The probability that I walk right the second time: 0.8
- It’s a product! $0.8^2$
Walking is a repetition of throws:

- A plan, e.g., \([Up, Up, Right, Right, Right]\), can bring us somewhere unintentionally

\[
\text{Plans and their value}
\]

\[
v_p(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right],
\]
the expected (discounted) sum of rewards.
Walking is a repetition of throws:

- A plan, e.g., \([\text{Up}, \text{Up}, \text{Right}, \text{Right}, \text{Right}]\), can bring us somewhere unintentionally
- How much is a plan worth?

\[
v_p(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right],
\]
the expected (discounted) sum of rewards.
Walking is a repetition of throws:

- A plan, e.g., \([Up, Up, Right, Right, Right]\), can bring us somewhere unintentionally.
- How much is a plan worth? 
\[
V^p(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right],
\]
the expected (discounted) sum of rewards.
Time and Risk

0.1

0.8

0.1

0.1

0.1

0.8

start
The real value of rewards depends on the agent’s patience. (as much as the real value of money depends on the attitude towards risk)
Time and Risk

Multiplicative discounting $\gamma^n$ after $n$ steps.
Time and Risk

Multiplicative discounting: $\gamma^n$ after $n$ steps.
Time and Risk

\[ \gamma = 0.5 \]
Time and Risk

\[ \gamma = 0.5 \]
Time and Risk

And now?

We include the probabilities...
And now?
We include the probabilities...
Probabilities of sequences:
to *discount* further the already discounted rewards!
Expected utility of this intended course of actions (not considering the rest = assuming it’s zero reward everywhere else) is: 6.9
Let’s see what happens if we go up instead...
Let’s see what happens if we go up instead...
Time and Risk

Including probabilities...
Time and Risk

Including probabilities...
Time and Risk

Summing up: 5.5
Time and Risk

This means that switching to $Up$ is dominated by going right.
This means that switching to *Up* is dominated by going right. Same reasoning for going down: lower expected utility!
Now I’m going to be very impatient.
\[ \gamma = 0.1 \]
Now I’m going to be very impatient.

\[ \gamma = 0.1 \]
Can you already see what’s going on?
Let's include the probabilities
Notice the impact of discounting on negative rewards: In the end, it’s all gonna be zero!
The expected utility at the starting state is: 1.428
Time and Risk

Let's go up
The expected utility at the starting state is: 1.806
Now Up is dominant!
A policy
A policy is a specification of moves at each decision point.
The expected utility (or value) of policy $\pi$, from state $s$ is:

$$v_\pi(s) = \mathbb{E}_{s_0} \left[ \sum_{t=0}^{\infty} \gamma^t r(S_t) \right]$$

where:

- $\mathbb{E}$, the probability distribution over the sequences is induced by:
  - the policy $\pi$ (the actions we are actually going to make)
  - the initial state $s_0$ (where we start)
  - the transition model (where we can get to)
Expected utility of a policy

The expected utility (or value) of policy $\pi$, from state $s$ is:

$$v^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right]$$
Expected utility of a policy

The expected utility (or value) of policy $\pi$, from state $s$ is:

$$v^\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right]$$

$E$, the probability distribution over the sequences is induced by:

- the policy $\pi$ (the actions we are actually going to make)
- the initial state $t$ (where we start)
- the transition model (where we can get to)
Expected utility of a policy

The expected utility (or value) of policy $\pi$, from state $s$ is:

$$v^\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(S_t)\right]$$

$E$, the probability distribution over the sequences is induced by:

- the policy $\pi$ (the actions we are actually going to make)
- the initial state $t$ (where we start)
- the transition model (where we can get to)
Optimal policies

If this was the entire (relevant) world... and $\gamma = 0$.

Going straight twice in a row would have value: $6.9$. 
If this was the entire (relevant) world...
If this was the entire (relevant) world... and $\gamma = 0.5$
If this was the entire (relevant) world...

and $\gamma = 0.5$

Going straight twice in a row would have value: 6.9
Optimal policies

We want the **optimal** policy:

\[ \pi^*_s = \arg\max_{\pi} v^\pi(s) \]

And we know that it’s unique no matter the starting state.
Optimal policies

If this was the entire (relevant) world...

and $\gamma = 0.5$.

Going straight twice in a row would be optimal.
Optimal policies

If this was the entire (relevant) world...
If this was the entire (relevant) world... and $\gamma = 0.5$
If this was the entire (relevant) world... and $\gamma = 0.5$
Going straight twice in a row would be optimal
Risk and reward

\[
\begin{align*}
    r &= [-\infty : -1.6284] \\
    r &= [-0.4278 : -0.0850] \\
    r &= [-0.0480 : -0.0274] \\
    r &= [-0.0218 : 0.0000]
\end{align*}
\]
The value of a state $s$ is its value under the optimal policy. In other words:

expected (discounted) sum of rewards assuming optimal actions
Value of states

6.9 is the value of the starting state.
Given the values of the states, choosing the best action is just MEU: maximize the expected utility of the immediate successors
Value of states

Figure: The values with $\gamma = 1$ and $R(s) = -0.04$
Value of states

Figure: The optimal policy

$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' | s, a) v(s')$

Maximise the expected utility of the subsequent state
Value of states

\[ \pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)v(s') \]

Maximise the expected utility of the subsequent state
The Bellman equation

Definition of utility of states leads to a simple relationship among values of neighboring states:
The Bellman equation

Definition of utility of states leads to a simple relationship among values of neighboring states:

Definition (Rewards)

$$\text{expected sum of rewards} = \text{current reward} + \gamma \times \text{expected sum of rewards after taking best action}$$
The Bellman equation

Bellman equation (1957):

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' | (s, a)) v(s') \]
The Bellman equation

Bellman equation (1957):

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' \mid (s, a)) v(s') \]

We can use it to compute the optimal policy!
Value Iteration Algorithm

1. Start with arbitrary values
2. Repeat for every $s$ simultaneously until “no change”

$$v(s) \leftarrow r(s) + \gamma \max_a \sum_{s'} v(s') P(s' | (s, a))$$
The Value Iteration Algorithm

\[
\textbf{Algorithm} : \text{VIA} \\
\text{Value Iteration}(MDP, \epsilon) \\
\textbf{Input}: \text{MDP, an MDP with states } S, \text{ actions } A(s), \text{ transition model } P(s' | s, a), \text{ rewards } R(s), \text{ discount } \gamma \\
\epsilon, \text{ the maximum error allowed in the utility of any state} \\
\textbf{Output}: \text{A utility function} \\
\text{begin} \\
\quad v \leftarrow v', \delta \leftarrow 0; /* \text{Using local variables to store information about values and value change */} \\
\text{while } \delta < \epsilon \left(\frac{1-\gamma}{\gamma}\right) \text{ do} \\
\quad \text{for each state } s \in S \text{ do} \\
\quad \quad v'[s] \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' | (s, a)) v'[s']; \\
\text{if } |v'[s] - v[s]| > \delta \text{ then} \\
\text{else } \delta \leftarrow |v'[s] - v[s]| \\
\text{return } v; \\
\text{end} \\
\text{end}
\]
A fundamental fact

Theorem

VIA:

- terminates
- returns the unique optimal policy (for the input values).
VIA in action
VIA in action
VIA in action

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Initialise the values, for $\gamma = 1, r = -0.04$
VIA in action

Simultaneously apply the Bellmann update to all states

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' \mid (s, a)) v(s') \]
Simultaneously apply the Bellmann update to all states

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' | (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' | (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' \mid (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' \mid (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' | (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' | (s, a)) v(s') \]
VIA in action

\[ v(s) = r(s) + \gamma \max_a \sum_{s'} P(s' \mid (s, a)) v(s') \]
Value Iteration Algorithm
The state values

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.812</td>
<td>0.868</td>
<td>0.912</td>
<td>+1</td>
</tr>
<tr>
<td>2</td>
<td>0.762</td>
<td>0.660</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.705</td>
<td>0.555</td>
<td>0.611</td>
<td>0.388</td>
</tr>
</tbody>
</table>
The optimal policy
Summary

- Stochastic actions can lead to unpredictable outcomes
- But we can still find optimal “strategies”, exploiting probabilities
What’s next

What if we don’t know what game we are playing?
What’s next

What if we don’t know what game we are playing?

Play anyway and see what happens!
What’s next

What if we don’t know what game we are playing?

Play anyway and see what happens!

and play as much as possible!