f

Reinforcement Learning

Basics

Value Function

We need to calculate the possible rewards for each action, $r^{avg}(a)$ to choose the correct action we want to perform.

The problem is, we have to calculate all the actions to figure out which action gives us the highest reward. But there is a better solution, we can use

the online version of the value function.

Offline Value Function

$$r^{avg}(a) = \frac{1}{N(a)} \sum_{i=1}^{n} r(a)_i = \frac{r_1 + r_2 \ldots + r_{N(a)}}{N(a)}$$

Create Online Value Funktion:

The following formulas show how to buid the online value function.

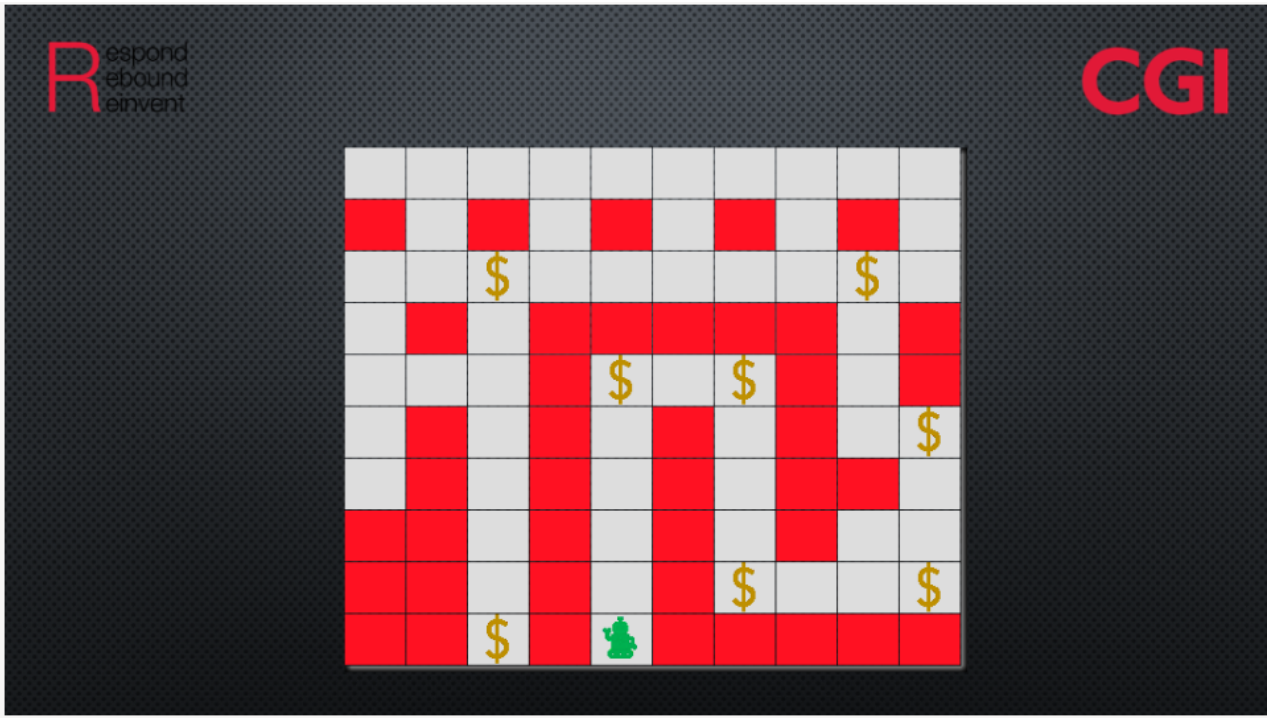
$$
\begin{aligned}
r_n^{avg} &= \frac{1}{N}\sum_{i=1}^{N} r_i \\
&= \frac{1}{N}\left(r_N + \sum_{i=1}^{N-1} r_i\right) \\
&= \frac{1}{N}\left(r_N + (N-1)\frac{1}{(N-1)}\sum_{i=1}^{(N-1)} r_i\right) \\
&= \frac{1}{N}\left(r_N + (N-1)r_{N-1}^{avg}\right) \\
&= \frac{1}{N}\left(r_N + Nr_{N-1}^{avg} - r_{N-1}^{avg}\right) \\
&= r_{N-1}^{avg} + \frac{1}{N}(r_N - r_{N-1}^{avg})
\end{aligned}
$$

What can we do with the online value function?

With the online value function we can calculate which actions has the best overall reward.

But what does us helps here, to find the best action for our robot?

Example for the second simulation



The possible actions are {**left, up, down, right**}. So we can use the **online value** function to calculate which of the actions are the best.

Example:

We estimate the average reward for the action "**up**":

$$r^{avg}(up) = \frac{r_1 + r_2 \ldots + r_{N(up)}}{N(up)}$$
$$=$$

One of the problems are, that we can't go with a dedicated action, because on a **given position** not every action is possible.

or try it with the **online** version:

$$r^{avg}(up) = r^{avg}_{N-1} + \frac{1}{N}(r_N - r^{avg}_{N-1})$$

Generell form of an exponentially weigthed moving average

$$r \leftarrow r + \alpha(r - r')$$

Old Formular:

$$Q(A) = Q(A) + \frac{1}{N(A)}[R - Q(A)]$$

New Formular:

$$r^{avg}(a) \leftarrow r^{avg}(a) + \frac{1}{N(a)}\Big[r - r^{avg}(a)\Big]$$

Example for new formular:

$$r^{avg}(up)_1 \leftarrow r^{avg}(up)_0 + \frac{1}{N(up)}\Big[r - r^{avg}(up)\Big]$$
$$\leftarrow$$

ϵ-greedy bandit algorithm in Julia

`function ϵ_greedy(ϵ)

```
if rand() <= ϵ
    move_agent_random()
  else
    move_max_action()
  end
r = checkreward()

r
end
```