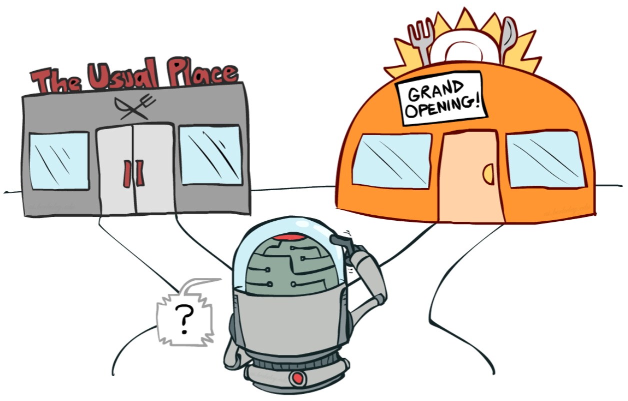
Exploration vs. Exploitation

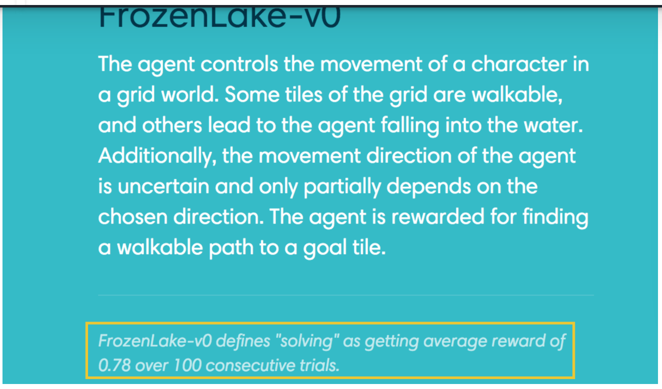


Exploration-Exploitation Dilemma
(Source)

Solving Environments in OpenAI Gym

In many cases, we would like our reinforcement learning (RL) agents to learn to maximize reward as quickly as possible. This can be seen in many OpenAI Gym environments.

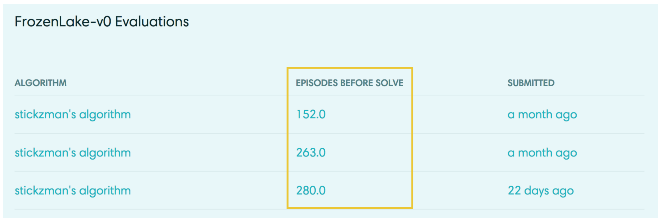
For instance, the FrozenLake-v0 environment is considered solved once the agent attains an average reward of 0.78 over 100 consecutive trials.


FrozenLake-v0

The agent controls the movement of a character in a grid world. Some tiles of the grid are walkable, and others lead to the agent falling into the water. Additionally, the movement direction of the agent is uncertain and only partially depends on the chosen direction. The agent is rewarded for finding a walkable path to a goal tile.

FrozenLake-v0 defines "solving" as getting average reward of 0.78 over 100 consecutive trials.

Algorithmic solutions to the FrozenLake-v0 environment are ranked according to the number of episodes needed to find the solution.


FrozenLake-v0 Evaluations

ALGORITHM	EPISODES BEFORE SOLVE	SUBMITTED
stickzman's algorithm	152.0	a month ago
stickzman's algorithm	263.0	a month ago
stickzman's algorithm	280.0	22 days ago

Solutions to Taxi-v1, Cartpole-v1, and MountainCar-v0 (along with many others) are also ranked according to the number of episodes before the solution is found. Towards this objective, it makes sense to design an algorithm that learns the optimal policy π_* as quickly as possible.

Exploration-Exploitation Dilemma

Recall that the environment's dynamics are initially unknown to the agent. Towards maximizing return, the agent must learn about the environment through interaction.

At every time step, when the agent selects an action, it bases its decision on past experience with the environment. And, towards minimizing the number of

Knowledge

devise a strategy where the agent always selects the action that it believes (*based on its past experience*) will maximize return. With this in mind, the agent could follow the policy that is greedy with respect to the action-value function estimate. We examined this approach in a previous video and saw that it can easily lead to convergence to a sub-optimal policy.

To see why this is the case, note that in early episodes, the agent's knowledge is quite limited (and potentially flawed). So, it is highly likely that actions *estimated* to be non-greedy by the agent are in fact better than the *estimated* greedy action.

With this in mind, a successful RL agent cannot act greedily at every time step (*that is*, it cannot always **exploit** its knowledge); instead, in order to discover the optimal policy, it has to continue to refine the estimated return for all state-action pairs (*in other words*, it has to continue to **explore** the range of possibilities by visiting every state-action pair). That said, the agent should always act *somewhat greedily*, towards its goal of maximizing return *as quickly as possible*. This motivated the idea of an ϵ-greedy policy.

We refer to the need to balance these two competing requirements as the **Exploration-Exploitation Dilemma**. One potential solution to this dilemma is implemented by gradually modifying the value of ϵ when constructing ϵ-greedy policies.

It makes sense for the agent to begin its interaction with the environment by favoring **exploration** over **exploitation**. After all, when the agent knows relatively little about the environment's dynamics, it should distrust its limited knowledge and **explore**, or try out various strategies for maximizing return. With this in mind, the best starting policy is the equiprobable random policy, as it is equally likely to explore all possible actions from each state. You discovered in the previous quiz that setting $\epsilon = 1$ yields an ϵ-greedy policy that is equivalent to the equiprobable random policy.

At later time steps, it makes sense to favor **exploitation** over **exploration**, where the policy gradually becomes more greedy with respect to the action-value function estimate. After all, the more the agent interacts with the environment, the more it can trust its estimated action-value function. You discovered in the previous quiz that setting $\epsilon = 0$ yields the greedy policy (or, the policy that most favors exploitation over exploration).

Thankfully, this strategy (of initially favoring exploration over exploitation, and then gradually preferring exploitation over exploration) can be demonstrated to be optimal.

Greedy in the Limit with Infinite Exploration (GLIE)

Knowledge

need to ensure that two conditions are met. We refer to these conditions as **Greedy in the Limit with Infinite Exploration (GLIE)**. In particular, if:

- every state-action pair s, a (for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$) is visited infinitely many times, and
- the policy converges to a policy that is greedy with respect to the action-value function estimate Q,

then MC control is guaranteed to converge to the optimal policy (in the limit as the algorithm is run for infinitely many episodes). These conditions ensure that:

- the agent continues to explore for all time steps, and
- the agent gradually exploits more (and explores less).

One way to satisfy these conditions is to modify the value of ϵ when specifying an ϵ-greedy policy. In particular, let ϵ_i correspond to the i-th time step. Then, both of these conditions are met if:

- $\epsilon_i > 0$ for all time steps i, and
- ϵ_i decays to zero in the limit as the time step i approaches infinity (that is, $\lim_{i\to\infty} \epsilon_i = 0$).

For example, to ensure convergence to the optimal policy, we could set $\epsilon_i = \frac{1}{i}$. (You are encouraged to verify that $\epsilon_i > 0$ for all i, and $\lim_{i\to\infty} \epsilon_i = 0$.)

Knowledge

As you read in the above section, in order to guarantee convergence, we must let ϵ_i decay in accordance with the GLIE conditions. But sometimes "guaranteed convergence" *isn't good enough* in practice, since this really doesn't tell you how long you have to wait! It is possible that you could need trillions of episodes to recover the optimal policy, for instance, and the "guaranteed convergence" would still be accurate!

> Even though convergence is **not** guaranteed by the mathematics, you can often get better results by either:
>
> - using fixed ϵ, or
> - letting ϵ_i decay to a small positive number, like 0.1.

This is because one has to be very careful with setting the decay rate for ϵ; letting it get too small too fast can be disastrous. If you get late in training and ϵ is really small, you pretty much want the agent to have already converged to the optimal policy, as it will take way too long otherwise for it to test out new actions!

As a famous example in practice, you can read more about how the value of ϵ was set in the famous DQN algorithm by reading the Methods section of the research paper:

> *The behavior policy during training was epsilon-greedy with epsilon annealed*

Knowledge

> *thereafter.*

When you implement your own algorithm for MC control later in this lesson, you are strongly encouraged to experiment with

NEXT

Knowledge