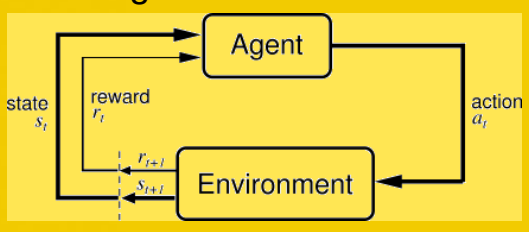
# INFINITE MARIO BROS

## APRENDIZAJE POR REFUERZO

**Ana Godoy Pérez**
**E.T.S.I Universidad de Huelva**
**Grado en Ingeniería Informática**

## INTRODUCCIÓN

El objetivo de este proyecto es crear un agente controlador para el juego Super Mario Bros basándose en el aprendizaje por refuerzo (reinforcement learning, RL). Este agente deberá aprender de su entorno para llegar lo más lejos posible en los distintos niveles, obteniendo la mejor puntuación posible en los mismos recolectando power-up y otros ítems y matando o esquivando enemigos.




Interacción agente-entorno en el aprendizaje por refuerzo.

## ALGORITMO EMPLEADO

El algoritmo empleado ha sido el Q-learning debido a su flexibilidad a la hora de aprender cómo comportarse correctamente por ser un algoritmo off-policy.
La **función de actualización** empleada por Q-learning es:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\,[R(s) + \gamma \max Q(s',a') - Q(s,a)]$$

La **política de selección** de acciones escogida ha sido la *ε-greedy*, tratando con ella de buscar un equilibrio relativo entre exploración y explotación.

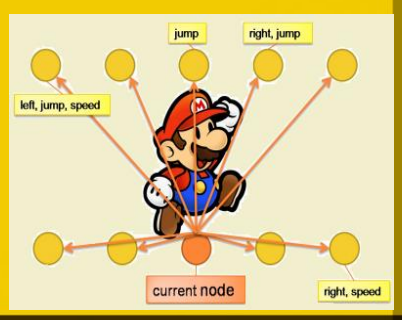
La tabla que contiene los valores que estiman la bondad de realizar una acción 'a' en un estado 's' (Q-table) se ha decidido inicializar a cero con el objetivo de ser lo más realista posible.

## ESTADOS Y ACCIONES

Los estados se han modelizado discretizando el entorno. Para ello se ha tenido en cuenta:

- El **modo** y el **estado** de Mario, así como si podía **saltar**, si estaba en el **suelo**, el número de **enemigos** con los que había **colisionado** y a los que había **matado** y la **distancia** que había recorrido desde el estado anterior.
- Los **enemigos** a su alrededor en un área de 4x4 celdas.
- Los **obstáculos** en frente de Mario que puede sobrepasar.

Posibles acciones a realizar por el agente




## EXPERIMENTACIÓN

En primer lugar se han tenido que determinar los parámetros más adecuados para el algoritmo. Una vez estos se han determinado, se ha procedido a "enseñar a Mario a jugar". En ambos casos se ha realizado una fase de entrenamiento y, a continuación, una fase de evaluación. La única diferencia entre el entrenamiento del primer y el segundo caso ha sido el número de iteraciones que ha realizado para entrenar (30.000 y 120.000 respectivamente).

**ENTRENAMIENTO:**

- Se ha entrenado el controlador con semillas y tipos de niveles aleatorios en cada iteración. Además cada partida la ha jugado en los tres modos en los que se puede encontrar.

**EVALUACIÓN:**

- Se ha evaluado el controlador jugando 1.000 partidas con semillas y tipos de niveles aleatorios. Sin embargo, el modo de inicio de Mario siempre es fire.

## RESULTADOS

Determinación de los parámetros:

| γ | α | ε | Distancia media recorrida (/256) | Partidas ganadas (%) |
|---|---|---|---|---|
| 0,5 | 0,1 | 0,3 | 104,909 | 235 |
| 0,5 | 0,2 | 0,3 | 162,564 | 308 |
| **0,6** | **0,25** | **0,3** | **202,93** | **329** |

Para determinar los parámetros se han realizado pruebas con todas las combinaciones para los siguientes valores: γ=[0.2 0.5 0.8] α=[0.1 0.2 0.3] ε=[0.3 0.6] Además se hicieron pruebas con otros parámetros coherentes con los resultados obtenidos. En la tabla superior se muestran los mejores resultados obtenidos.

Finalmente, con los parámetros elegidos (γ=0,6 α=0,25 ε=0,3) realizó un entrenamiento de 120,000 iteraciones con una posterior etapa de evaluación en las que **el agente superó el 61,9% de las partidas.**

## CONCLUSIONES

- A la hora de dotar de inteligencia al agente tiene una mayor influencia el valor de los rewards que la complejidad del estado.
- Es importante explorar todas las posibles acciones para un estado porque podemos estar pasando por alto la acción más adecuada.