UNIVERSIDAD ALFONSO X EL SABIO

Business Tech

Máster Universitario en Inteligencia Artificial

TRABAJO DE FIN DE MÁSTER

Dinámicas de Cooperación en Sistemas Multiagente: Un Enfoque Basado en Aprendizaje por Refuerzo

Samuel Lozano Iglesias

Director: Juan Agustín Fraile Nieto

Junio de 2025

Resumen

Este Trabajo de Fin de Máster analiza la emergencia de comportamientos cooperativos en sistemas multiagente utilizando técnicas de Aprendizaje por Refuerzo Multiagente (MARL). Se desarrolla un entorno simulado, *CoopCoins*, que permite modelar dilemas sociales en contextos de motivación mixta. La novedad del enfoque radica en la incorporación de actitudes sociales como sesgos internos que modulan el aprendizaje de los agentes. Mediante el análisis de métricas actitudinales (coherencia, resiliencia y fuerza), se estudia la compatibilidad entre perfiles, la influencia mutua y la estabilidad de las interacciones. Los resultados muestran que determinadas configuraciones actitudinales favorecen la cooperación espontánea, mientras que otras promueven dinámicas destructivas. Este trabajo aporta una base empírica para el diseño de agentes artificiales socialmente sensibles en entornos complejos.

Palabras clave: Aprendizaje por refuerzo multiagente, teoría de juegos, dilemas sociales, actitud.

Abstract

This Master Thesis analyses the emergence of cooperative behaviours in multi-agent systems using Multi-Agent Reinforcement Learning (MARL) techniques. A simulated environment, *CoopCoins*, is developed to model social dilemmas in mixed motivation contexts. The novelty of the approach lies in the incorporation of social attitudes as internal biases that modulate agents' learning. By analysing attitudinal metrics (coherence, resilience and strength), compatibility between profiles, mutual influence and stability of interactions are studied. The results show that certain attitudinal configurations favour spontaneous cooperation, while others promote destructive dynamics. This work provides an empirical basis for the design of socially responsive artificial agents in complex environments.

Keywords: Multiagent reinforcement learning, game theory, social dilemmas, attitude.

Índice general

CAPÍTULO 1

Introducción

En nuestro mundo, la cooperación no es un accidente. Desde las bacterias que forman colonias hasta las aves que migran en formación, pasando por comunidades humanas que construyen civilizaciones sobre la base del entendimiento mutuo, la cooperación emerge como una solución evolutiva a la escasez, al riesgo y a la incertidumbre. Pero, ¿qué ocurre cuando trasladamos esta pregunta a un terreno artificial? ¿Pueden los agentes artificiales, entrenados para maximizar recompensas individuales, aprender por sí mismos a colaborar? ¿Y si además comparten espacio con otros agentes egoístas, impredecibles o incluso hostiles?

Este Trabajo de Fin de Máster se sumerge en esta cuestión desde una perspectiva experimental, combinando los marcos teóricos del Aprendizaje por Refuerzo Multiagente (MARL) y la teoría de juegos, con la ambición de desentrañar los mecanismos que describen la emergencia de comportamientos colectivos —como la cooperación— en sistemas compuestos por múltiples agentes autónomos. En particular, se propone una aproximación novedosa que introduce el concepto de **actitud** como sesgo interno que modula el aprendizaje de cada agente, permitiendo estudiar no solo qué aprenden los agentes, sino cómo lo hacen, y con qué propensión social.

Motivación y contexto

En los últimos años, el estudio del comportamiento colectivo en sistemas artificiales ha adquirido una importancia creciente. Aplicaciones como la coordinación de robots, la gestión distribuida de recursos o la toma de decisiones en redes autónomas requieren no solo que los agentes sean inteligentes, sino que sean capaces de interactuar con otros agentes en contextos complejos, inciertos y muchas veces con-

flictivos. Tradicionalmente, se han planteado soluciones centralizadas, mecanismos de control explícito o el diseño de recompensas cuidadosamente afinadas para inducir comportamientos deseados. Sin embargo, en muchos entornos reales, la intervención centralizada no es viable o directamente no existe.

Aquí es donde el MARL se revela como una herramienta poderosa. Permite entrenar agentes mediante la experiencia directa y la exploración en el entorno, sin necesidad de modelar todos los aspectos de la interacción. No obstante, el salto del aprendizaje de un único agente al caso multiagente implica desafíos significativos: la no estacionariedad del entorno, la asignación de recompensas, la exploración conjunta o el equilibrio entre competencia y cooperación.

Más allá del rendimiento, este trabajo se interesa por una cuestión más profunda: *¿pueden los agentes artificiales desarrollar estructuras sociales estables sin ser explícitamente programados para ello?* Y, si la respuesta es afirmativa, ¿bajo qué condiciones? ¿Qué factores del entorno, del entrenamiento o de la configuración interna de los agentes favorecen o dificultan este proceso?

Objetivos

El objetivo central de este trabajo es explorar, desde un enfoque computacional y cuantitativo, las condiciones que permiten la emergencia espontánea de la cooperación en sistemas multiagente. Para ello, se han definido los siguientes objetivos específicos:

- **Diseñar un entorno simulado**, denominado *CoopCoins*, capaz de inducir dinámicas sociales complejas mediante la presencia de recursos compartidos, recompensas locales y la inclusión explícita de dilemas sociales como el del prisionero.
- **Incorporar actitudes sociales** a los agentes, es decir, sesgos que alteran su proceso de aprendizaje en función de una predisposición cooperativa, individualista, competitiva, etc.
- **Analizar el comportamiento emergente** de los agentes en función de sus actitudes y del contexto social en el que se desenvuelven, utilizando métricas como la coherencia, la resiliencia o la fuerza actitudinal.
- **Estudiar el impacto de los dilemas sociales** sobre la dinámica del sistema, evaluando si su introducción fomenta o destruye la cooperación, y cómo este efecto depende de las combinaciones de actitudes presentes.

El propósito último de esta investigación no es simplemente obtener agentes que cooperen más, sino entender qué significa *cooperar* en un contexto de aprendizaje automático. Dar respuesta a cómo surgen los patrones sociales a partir de simples reglas individuales, y cómo diseñar sistemas artificiales en los que emerja espontáneamente una dinámica social deseable.

Estrategia y estructura del trabajo

Para alcanzar estos objetivos, se ha desarrollado una arquitectura experimental articulada en el entorno *CoopCoins*, una simulación interactiva en la que dos agentes compiten y cooperan por recoger monedas. Cada agente puede ser entrenado con una actitud determinada y, además, puede modificar su comportamiento en función del perfil de su oponente. Esto permite estudiar la interacción social como un fenómeno dinámico y contextual, más allá de simples heurísticas fijas.

El entorno simula dilemas sociales secuenciales y explora distintas configuraciones de entrenamiento, tanto centralizadas como descentralizadas. Se analizan combinaciones de actitudes, se introducen métricas actitudinales y se examina el efecto de los dilemas sobre la estabilidad del comportamiento emergente.

La organización del trabajo refleja este enfoque progresivo:

- En el **Capítulo 2** se introduce el marco teórico del aprendizaje por refuerzo, desde el caso de un único agente hasta las extensiones multiagente, incluyendo aspectos técnicos y arquitecturas relevantes.
- En el **Capítulo 3** se presentan los fundamentos de la teoría de juegos y los dilemas sociales más relevantes, con especial atención al dilema del prisionero y su generalización a contextos secuenciales.
- El **Capítulo 4** repasa el estado del arte en MARL, comparando diferentes dinámicas de entrenamiento y resaltando estudios previos sobre emergencia de la cooperación.
- En el **Capítulo 5** se describe detalladamente la metodología: el entorno desarrollado *CoopCoins*, la implementación técnica, las actitudes diseñadas, el sistema de recompensas y las métricas empleadas. Además, se incluye el plan de trabajo a seguir.
- En el **Capítulo 6** se presentan y analizan los resultados empíricos, explorando las interacciones actitudinales, la evolución del comportamiento de los agentes, el efecto del dilema social y la caracterización de actitudes robustas.
- Finalmente, el **Capítulo 7** ofrece una reflexión sobre los resultados del trabajo y las líneas abiertas para futuras investigaciones.

Una nota final

Este trabajo no pretende resolver todos los problemas abiertos del aprendizaje multiagente, ni ofrecer una receta universal para inducir comportamientos en entornos artificiales. En cambio, aspira a hacer visible un fenómeno sutil pero esencial: que la cooperación no siempre necesita ser impuesta, sino que puede emerger, adaptarse y evolucionar si se siembran las condiciones adecuadas. Y que, tal vez, entender cómo aprenden a cooperar nuestros algoritmos no solo nos ayude a construir mejores sistemas, sino también a entendernos mejor a nosotros mismos.

CAPÍTULO 2

Aprendizaje por refuerzo

El Aprendizaje por Refuerzo Multiagente (MARL, por sus siglas en inglés) estudia sistemas compuestos por múltiples agentes —como robots o vehículos— que interactúan dentro de un entorno común. En cada instante de tiempo, cada agente toma acciones de manera autónoma en función del estado en el que se encuentra el entorno. El propósito de los algoritmos MARL es que cada agente aprenda una *política*, i.e. un conjunto de acciones asociadas a cada estado, que permita alcanzar la máxima recompensa posible. Esta recompensa está asociada al objetivo que se busca conseguir, e.g. una recompensa por completar una tarea.

Estos agentes son entidades capaces de *aprender*, entendiendo este aprendizaje como un ajuste en sus comportamientos a partir de la experiencia adquirida al interactuar con el entorno. El desafío principal entonces radica en la alta complejidad de los entornos y la naturaleza combinatoria de los problemas abordados: muchos de los problemas que se abordan en MARL están clasificados como problemas NP-complejos[1].

Dado que el aprendizaje en entornos multiagente introduce una serie de complejidades adicionales, como la no estacionariedad, la coordinación entre agentes y la escalabilidad, se comenzará el análisis teórico a partir del marco del aprendizaje por refuerzo en entornos de un único agente. Este enfoque permite sentar las bases conceptuales necesarias para, posteriormente, adentrarse en el estudio de sistemas con múltiples agentes que cooperan entre sí para lograr objetivos compartidos o complementarios.

[1]Un problema NP-comlpejo es aquel para el cual no se conoce un algoritmo que lo resuelva en tiempo polinomial, y cuya solución no necesariamente puede verificarse en tiempo polinomial.

2.1. Aprendizaje por refuerzo de un único agente

El aprendizaje por refuerzo en entornos de un solo agente se basa en la interacción continua entre un agente y su entorno. En este esquema, el agente toma decisiones eligiendo acciones, y a cambio recibe información sobre el nuevo estado del entorno y una señal de recompensa, que refleja si su acción ha sido beneficiosa o no con respecto a un objetivo determinado (véase la figura 2.1). El propósito principal del agente es aprender una estrategia de comportamiento que le permita maximizar una medida acumulada de recompensas a lo largo del tiempo.

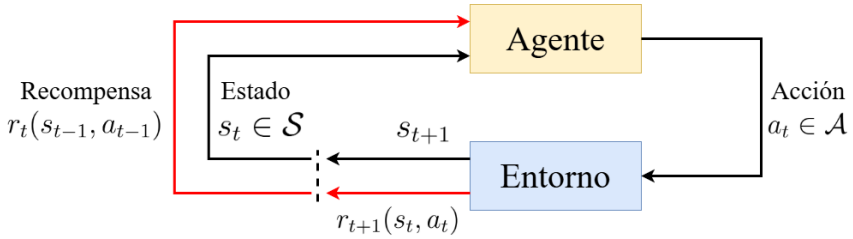


Figura 2.1: *Esquema general del aprendizaje por refuerzo en un entorno de un solo agente. El agente observa el estado del entorno $s_t \in \mathcal{S}$, selecciona una acción $a_t \in \mathcal{S}$, recibe una recompensa $r_{t+1} \in \mathbb{R}$ y una nueva observación del estado $s_{t+1} \in \mathcal{S}$.*

Este proceso de toma de decisiones puede formalizarse mediante los denominados Procesos de Decisión de Markov (*Markov Decision Processes*, MDP), que constituyen la base matemática sobre la cual se construyen los algoritmos de aprendizaje por refuerzo [5]. A partir de esta formulación, es posible desarrollar soluciones que permiten al agente aprender de la experiencia, ajustando su comportamiento en función de las consecuencias observadas tras cada acción.

Definición 2.1.1. *Un Proceso de Decisión de Markov (Markov Decision Process, MDP) se define como una cuádrupla $(\mathcal{S}, \mathcal{A}, p, r)$, donde:*

- *$\mathcal{S}$ es el conjunto de estados.*
- *$\mathcal{A}$ es el conjunto de acciones.*
- *$p(s'|s, a)$ es la función de transición, que indica la probabilidad de que el entorno pase del estado s al estado s' al ejecutar la acción a. Se cumple entonces que $\sum_{s' \in \mathcal{S}} p(s'|s, a) = 1$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$.*
- *$r(s, a, s')$ es la función que representa la recompensa instantánea esperada que el agente obtiene al realizar la acción a en el estado s, pasando al estado s'.*

En resumen, en cada tiempo t el agente observa un estado $s_t \in \mathcal{S}$, decide tomar la acción $a_t \in \mathcal{A}(s_t) \subseteq \mathcal{A}$ perteneciente al conjunto de acciones válidas para el estado s_t, y recibe una recompensa inmediata $r(s_t, a_t, s_{t+1}) \in \mathbb{R}$. El entorno transita a un nuevo estado $s_{t+1} \in \mathcal{S}$, de acuerdo con una distribución de probabilidad $p(s_{t+1}|s_t, a_t)$. Una propiedad clave es que tanto la función de transición como la función de recompensa dependen exclusivamente de la acción actual y los estados predecesor y sucesor, sin necesidad de considerar el historial completo del proceso.

Hay varias cuestiones de notación y definiciones adicionales que deben tenerse en cuenta para analizar adecuadamente los procesos de decisión de Markov:

- Se reservan las letras mayúsculas A_t, S_t para denotar las variables aleatorias relacionadas con la acción y el estado, mientras que las letras minúsculas denotan las acciones y estados concretos.
- Por simplicidad, el conjunto de acciones posibles en un estado dado s_t, $\mathcal{A}(s_t)$, se va a considerar siempre igual al conjunto de acciones total, i.e. $\mathcal{A}(s_t) = \mathcal{A}$, $\forall s_t \in \mathcal{S}$.
- En ocasiones resulta conveniente utilizar una función de recompensa esperada asociada a los estados y acciones actuales, i.e. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, sin que dependa del estado siguiente. Esta función se define como:

$$r(s, a) = \sum_{s' \in \mathcal{S}} p(s'|s, a) \cdot r(s, a, s'). \tag{2.1}$$

- Suele introducirse un quinto elemento $\gamma \in [0, 1]$ en la tupla de un proceso de decisión de Markov, denominado factor de descuento, que determina la importancia relativa de las recompensas futuras frente a las inmediatas.

El objetivo del agente es encontrar una política $\pi : \mathcal{S} \to \mathcal{A}$, que maximice la recompensa acumulada J. En general, las políticas pueden ser deterministas, como sería el caso de una política $\pi : \mathcal{S} \to \mathcal{A}$ que asocia estados con acciones, o estocástica, en la que no siempre para cada estado se toma la misma acción.

Para representar correctamente esta distinción, en general se considera la política como una función $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ que representa la probabilidad de que la acción $a \in \mathcal{A}$ sea tomada si se considera el estado $s \in \mathcal{S}$. Así, la política se denota como $\pi(a|s)$, donde el símbolo "|" solo indica que la acción está condicionada por el estado. En definitiva:

$$\pi(a|s) := \pi(s, a) = \Pr\left(A_t = a | S_t = s\right), \ \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{2.2}$$

Con esta nueva representación, es sencillo observar que las políticas deterministas serán de la forma $\pi : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$, y las políticas estocásticas serán de la forma $\pi : \mathcal{S} \times \mathcal{A} \to (0, 1)$.

Por otro lado, la recompensa acumulada J depende únicamente de la política π, del estado inicial del entorno s_0 y del factor de descuento $\gamma \in [0, 1]$, que mide cuánto afectan las recompensas futuras con respecto a las actuales:

$$J = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r(S_t, A_t, S_{t+1}) \mid A_t \sim \pi(\cdot \mid S_t),\ S_0 = s_0\right]. \tag{2.3}$$

Observación 2.1.2. *Si se considera $\gamma = 1$, entonces es requisito para la convergencia del problema que exista un tiempo máximo T en el que termine el proceso de decisión, y el sumatorio sería de $t = 0$ hasta dicho T. En caso contrario, la recompensa acumulada podría divergir. Cuando existe un tiempo máximo T acotado se dice que se considera el "caso episódico".*

Bajo una política dada π, se definen la función de valor partiendo del estado s, denotada como $V_\pi(s)$:

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r(S_t, A_t, S_{t+1}) \mid A_t \sim \pi(\cdot \mid S_t),\ S_0 = s\right], \tag{2.4}$$

y la función de acción-valor (*Q-valor*), partiendo del estado s y la acción a:

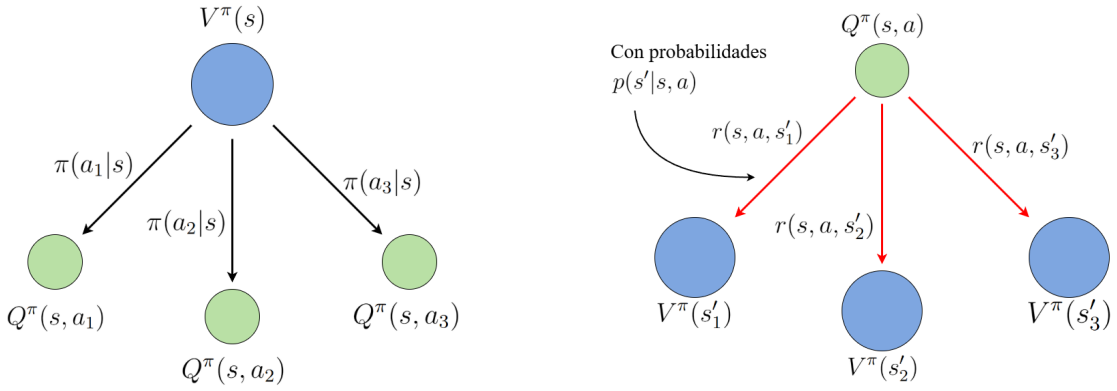
$$Q_\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r(S_t, A_t, S_{t+1}) \mid A_t \sim \pi(\cdot \mid S_t),\ S_0 = s, A_0 = a\right]. \tag{2.5}$$

Cabe observar que las funciones de valor y de acción-valor están muy relacionadas. De hecho, la función de valor asociada a una política π y un estado s no es más que la suma sobre todas las acciones de la función de acción-valor, ponderada por la probabilidad de seleccionar dicha acción, i.e. ponderada por la propia política (véase la figura 2.2a):

$$V_\pi(s) = \sum_{a \in \mathcal{A}} \pi(s, a) \cdot Q_\pi(s, a) =: \sum_{a \in \mathcal{A}} \pi(a|s) \cdot Q_\pi(s, a). \tag{2.6}$$

En el otro sentido, la función de acción-valor asociada a una política π, en un estado s y una acción a, no es más que la recompensa inmediata actual, independientemente del estado siguiente, i.e. ecuación (2.1), más las funciones de valor de los posibles estados siguientes, ponderados por el factor de descuento γ y la función de transición (véase la figura 2.2b):

$$Q_\pi(s, a) = \sum_{s' \in \mathcal{S}} p(s'|s, a) \cdot [r(s, a, s') + \gamma \cdot V_\pi(s')] = r(s, a) + \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s, a) \cdot V_\pi(s'). \tag{2.7}$$



(a) *Relación de la función de valor $V_\pi(s)$ con la función de acción-valor $Q_\pi(s, a)$ a través de la política π.*

(b) *Relación de la función de acción-valor $Q_\pi(s, a)$ con la función de valor $V_\pi(s')$, la recompensa y la función de transición.*

Figura 2.2: *Relación bidireccional entre las funciones de valor y de acción-valor bajo una política dada.*

Por último, en base a esta descripción, los algoritmos de aprendizaje por refuerzo suelen organizarse en torno a dos enfoques principales: la iteración sobre valores (*value iteration*) y la iteración sobre políticas (*policy iteration*).

En el enfoque de iteración sobre valores, el objetivo es aproximar una función de valor óptima ($V_{\pi^*}(s)$) o una función acción-valor óptima ($Q_{\pi^*}(s,a)$), para después extraer una política óptima seleccionando las acciones que maximizan los valores esperados. Este enfoque se basa en la idea de que, al conocer qué tan bueno es un estado o acción en términos de recompensas futuras, es posible guiar el comportamiento del agente de manera eficiente.

Por otro lado, el enfoque de iteración sobre políticas se centra directamente en optimizar la política sin necesidad de estimar explícitamente funciones de valor. Aquí, el agente explora el espacio de políticas posibles y las ajusta gradualmente con el objetivo de maximizar la recompensa esperada. Este método suele involucrar técnicas de gradiente, como el *policy gradient*, que permiten actualizar los parámetros de la política de manera continua a partir de la experiencia acumulada.

Ambos paradigmas representan distintas estrategias para resolver el mismo problema fundamental: encontrar una política que permita al agente tomar decisiones óptimas en un entorno incierto y dinámico.

2.1.1. Métodos basados en el valor

Los métodos basados en el valor tienen como objetivo principal estimar una función de valor óptima, como la función acción-valor $Q^*(s,a)$, que permite al agente seleccionar acciones que maximicen la recompensa esperada acumulada. Una vez aproximada dicha función, se deriva la política óptima eligiendo la acción con el mayor valor estimado para cada estado:

$$\pi^*(s) = \arg\max_{a \in A} Q^*(s,a). \tag{2.8}$$

De esta forma, la derivación de la política se realiza a partir de la función acción-valor sobre la que se itera, pero podría ser que la iteración de $Q(s,a)$ dependiera a su vez de la política. Este es el caso de los *on-policy TD methods*, i.e. métodos de diferencia temporal utilizando la política, donde destaca el método *Sarsa* [50].

La idea principal en estos métodos es que se estima $Q_\pi(s,a)$ utilizando la política π, y se cambia la política π utilizando la función $Q_\pi(s,a)$ actualizada. El algoritmo de actualización de *Sarsa* utiliza la quintupla completa $(s_t, a_t, r, s_{t+1}, a_{t+1})$, de ahí el nombre, y es:

$$Q_\pi(s_t,a_t) \leftarrow Q_\pi(s_t,a_t) + \alpha \cdot \left[r(s_t,a_t,s_{t+1}) + \gamma \cdot Q_\pi(s_{t+1},a_{t+1}) - Q_\pi(s_t,a_t)\right], \tag{2.9}$$

donde α es la tasa de aprendizaje, γ es el factor de descuento, y el término entre corchetes representa el error temporal (*TD-error*).

Para evitar esta dependencia del algoritmo de actualización con la política π, se han desarrollado los *off-policy TD methods*, i.e. métodos de diferencia temporal que no utilizan la política. Uno de los algoritmos más representativos en este paradigma es *Q-learning* [67]. Este algoritmo actualiza iterativamente una estimación de la función Q según la siguiente regla [68]:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha \left[r(s_t,a_t,s_{t+1}) + \gamma \cdot \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right]. \tag{2.10}$$

Observación 2.1.3. *Bajo ciertas condiciones generales, la función Q obtenida con Q-learning converge a la función de acción-valor óptima Q^* [40], independientemente de la política utilizada.*

Deep Q-Networks (DQN)

En general, los espacios de estados y/o acciones son grandes, resultando imposible obtener de manera exacta las funciones $Q(s,a)$ o $V(s)$ por la conocida *maldición de la dimensión*[2]. Por ello, en la práctica se emplean aproximadores de funciones, tanto lineales como no lineales, para representar la función de acción-valor $Q(s,a)$.

Esto es, se aproxima la función óptima Q^* a partir de una función $Q_{\boldsymbol{\theta}}(s,a)$ dependiente de unos parámetros $\boldsymbol{\theta} \in \mathbb{R}^d$ desconocidos, que se aprenden mediante técnicas de aprendizaje supervisado. Si bien los métodos que utilizan funciones de aproximación lineales garantizan convergencia local [6], las redes neuronales permiten capturar relaciones no lineales más complejas y han demostrado alto desempeño empírico [41].

Con la introducción de estas redes neuronales profundas, se propuso *Deep Q-Learning* (DQN) para aproximar la función $Q_{\boldsymbol{\theta}}(s,a)$, dependiente de los parámetros $\boldsymbol{\theta} \in \mathbb{R}^d$ [42]. Para ello, se minimiza la siguiente función de pérdida, basada en el *TD-error*:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[(y - Q_{\boldsymbol{\theta}}(s,a))^2\right], \quad \text{donde} \quad y = r + \gamma \max_{a'} Q_{\boldsymbol{\theta}^-}(s',a'). \tag{2.11}$$

Aquí, $Q_{\boldsymbol{\theta}^-}$ es una red objetivo que se actualiza periódicamente para estabilizar el aprendizaje. Además, se utiliza una memoria de experiencias $\mathcal{D}$ (*experience replay buffer*) [31] para romper la correlación temporal entre muestras consecutivas. En definitiva, cuando la red neuronal se ha entrenado correctamente, permite obtener la función de acción-valor para todas las acciones sin tener que iterar sobre ellas, obteniendo así la política aprendida (véase la figura 2.3).

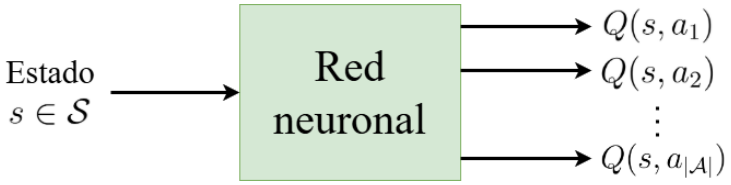


Figura 2.3: *Esquema del algoritmo DQN, que utiliza una red neuronal entrenada para obtener la función de acción-valor.*

2.1.2. Métodos basados en la política

Los métodos basados en la política representan otra clase de algoritmos de aprendizaje por refuerzo (RL) que, a diferencia de los métodos basados en el valor, buscan

[2]La maldición de la dimensión, o *curse of dimensionality* en inglés, se refiere al crecimiento exponencial del espacio de estados al aumentar la dimensión del problema, imposibilitando tanto el almacenamiento como el análisis exacto del problema.

aproximar directamente la política óptima $\pi^*(s)$ mediante una política $\pi_{\boldsymbol{\theta}}(s)$ parametrizada por $\boldsymbol{\theta} \in \mathbb{R}^d$. El objetivo, entonces, es maximizar la recompensa esperado $J(\boldsymbol{\theta})$ ajustando los parámetros de la política mediante técnicas de optimización basadas en gradientes. Esta recompensa esperada, o rendimiento, se suele estimar como la función de valor partiendo del estado inicial: $J(\boldsymbol{\theta}) = V_{\pi_{\boldsymbol{\theta}}}(s_0)$. Así, la recompensa esperada es:

$$J(\boldsymbol{\theta}) = V_{\pi_{\boldsymbol{\theta}}}(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r(S_t, A_t, S_{t+1}) \mid A_t \sim \pi_{\boldsymbol{\theta}}(\cdot \mid S_t),\ S_0 = s_0\right]. \tag{2.12}$$

De esta manera, se pueden aprender directamente políticas estocásticas cuando la política óptima no es determinista, además de permitir una convergencia más estable y suave cuando se usan redes neuronales profundas como aproximadores de funciones [33, 66].

Los *policy gradient methods* proporcionan una forma de estimar el gradiente del rendimiento con respecto a los parámetros de la política, de forma que la actualización de los parámetros que determinan la política se hace mediante una tasa de aprendizaje α que maximiza la recompensa acumulada:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \cdot \nabla J(\boldsymbol{\theta}_t). \tag{2.13}$$

El cálculo del gradiente $\nabla J(\boldsymbol{\theta}_t)$ en los distintos métodos se basa en el *policy gradient theorem* [60]. Para enunciarlo y demostrarlo, primero se debe definir la distribución de estados $\mu(s)$, representando la probabilidad de encontrar cada uno de los estados si se utiliza la política $\pi_{\boldsymbol{\theta}}$:

Definición 2.1.4. *La distribución de estados en la política, on-policy state distribution en inglés, $\mu_{\pi_{\boldsymbol{\theta}}}(s)$ representa la probabilidad de encontrar cada uno de los estados $s \in \mathcal{S}$ si se utiliza la política $\pi_{\boldsymbol{\theta}}$. Esto es:*

$$\mu_{\pi_{\boldsymbol{\theta}}}(s) = \frac{\sum_{t=0}^{\infty} \gamma^t \cdot Pr(s_0 \to s, t, \pi_{\boldsymbol{\theta}})}{\sum_{s' \in \mathcal{S}} \sum_{t=0}^{\infty} \gamma^t \cdot Pr(s_0 \to s', t, \pi_{\boldsymbol{\theta}})}, \tag{2.14}$$

donde $Pr(s_0 \to s, t, \pi_{\boldsymbol{\theta}})$ es la probabilidad de alcanzar el estado s desde el estado inicial s_0 en t unidades de tiempo.

En ocasiones, esta distribución se define simplemente como la fracción de tiempo dedicado al estado s [60]. Además, la distribución de estados de la política cumple que $\mu_{\pi_{\boldsymbol{\theta}}}(s) \geq 0$, $\forall s \in \mathcal{S}$ y $\sum_{s \in \mathcal{S}} \mu_{\pi_{\boldsymbol{\theta}}}(s) = 1$.

Teorema 2.1.5. *Para cualquier proceso de decisión de Markov con las condiciones descritas en la definición 2.1.1, siendo la política $\pi_{\boldsymbol{\theta}} : \mathcal{A} \times \mathcal{S} \to [0, 1]$, dependiente de unos parámetros $\boldsymbol{\theta} \in \mathbb{R}^d$ con $d < |\mathcal{S}|$, y considerando la distribución de estados de la política $\mu_{\pi_{\boldsymbol{\theta}}}(s)$ dada por la definición 2.1.4. Entonces, si la política es diferenciable con respecto a dichos parámetros, el gradiente de la recompensa acumulada $J(\theta)$, dada por la ecuación (2.12), es:*

$$\nabla J(\boldsymbol{\theta}) \propto \sum_{s \in \mathcal{S}} \mu_{\pi_{\boldsymbol{\theta}}}(s) \sum_{a \in \mathcal{A}} Q_{\pi_{\boldsymbol{\theta}}}(s, a) \cdot \nabla \pi_{\boldsymbol{\theta}}(s, a), \tag{2.15}$$

donde $Q_{\pi_\theta}(s, a)$ es la función de acción-valor.

Demostración. Véase el teorema 8.1.1, en el capítulo de apéndices. □

Observación 2.1.6. *La relación* (2.15) *presenta un término de proporcionalidad que es la longitud promedio del episodio en el caso episódico, y que es 1 (siendo, de hecho, una igualdad) en el caso continuo en el que* $T \to \infty$ *[60].*

Algoritmo REINFORCE

Considerando la ecuación (2.15) del *policy gradient theorem*, se observa que es una suma sobre el conjunto de estados posibles. Considerando como variable aleatoria S_t el estado escogido, esta suma no es más que el valor esperado, luego:

$$\nabla J(\boldsymbol{\theta}) \propto \sum_{s\in\mathcal{S}} \mu_{\pi_{\boldsymbol{\theta}}}(s) \sum_{a\in\mathcal{A}} \nabla\pi_{\boldsymbol{\theta}}(x,a)\cdot Q_{\pi_{\boldsymbol{\theta}}}(x,a) = \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{a\in\mathcal{A}} \nabla\pi_{\boldsymbol{\theta}}(S_t,a)\cdot Q_{\pi_{\boldsymbol{\theta}}}(S_t,a)\right].$$

Ahora, introduciendo la variable aleatoria A_t que representa la acción elegida y añadiendo un pesado sobre $\pi_{\boldsymbol{\theta}}(S_t,a)$ para poder realizar el valor esperado:

$$\begin{aligned}
\nabla J(\boldsymbol{\theta}) &\propto \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{a\in\mathcal{A}} \frac{\nabla\pi_{\boldsymbol{\theta}}(S_t,a)}{\pi_{\boldsymbol{\theta}}(S_t,a)}\cdot Q_{\pi_{\boldsymbol{\theta}}}(S_t,a)\cdot\pi_{\boldsymbol{\theta}}(S_t,a)\right] = \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\frac{\nabla\pi_{\boldsymbol{\theta}}(S_t,A_t)}{\pi_{\boldsymbol{\theta}}(S_t,A_t)}\cdot Q_{\pi_{\boldsymbol{\theta}}}(S_t,A_t)\right] = \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\frac{\nabla\pi_{\boldsymbol{\theta}}(S_t,A_t)}{\pi_{\boldsymbol{\theta}}(S_t,A_t)}\cdot \sum_{k=t}^{T}\gamma^{k-t}\cdot r(S_k,A_k,S_{k+1})\right] = \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\nabla\log\left(\pi_{\boldsymbol{\theta}}(S_t,A_t)\right)\cdot \sum_{k=t}^{T}\gamma^{k-t}\cdot r(S_k,A_k,S_{k+1})\right],
\end{aligned}$$

donde en el penúltimo paso se ha considerado que, por definición:

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{k=t}^{T}\gamma^{k-t}\cdot r(S_k,A_k,S_{k+1})\right] =: Q_{\pi_{\boldsymbol{\theta}}}(S_t,A_t).$$

De esta forma, el algoritmo REINFORCE [69] estima el gradiente utilizando el retorno observado y la actualización de los parámetros se realiza como:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\cdot\sum_{k=t}^{T}\gamma^{k-t}\cdot r(s_k,a_k,s_{k+1})\cdot\nabla\log\left(\pi_{\boldsymbol{\theta}_t}(s_t,a_t)\right). \tag{2.16}$$

Esta actualización supone la base del algoritmo REINFORCE, y al utilizar la recompensa completa, i.e. para todos los tiempos futuros, en cada iteración, se cataloga dentro del conjunto de métodos de Monte Carlo[3]. Por ende, y como le ocurre

[3]Los métodos de Monte Carlo son algoritmos que utilizan muestreo aleatorio para aproximar soluciones numéricas a problemas complejos. Se caracterizan por: (1) el uso de aleatoriedad, (2) la repetición de simulaciones, (3) la estimación de resultados mediante promedios estadísticos, y (4) su aplicabilidad cuando no se dispone de una solución analítica o exacta.

a todos estos métodos, el algoritmo sufre de una alta varianza que ralentiza el entrenamiento.

Para reducir la varianza asociada, se puede introducir un término auxiliar $b(S_t)$, denominado *baseline*, que depende del estado pero no de la acción [69]. Este término, como su nombre indica, ejerce como una referencia a la que se debe aproximar $Q_\pi(s,a)$, por lo que se introduce restándose a este término en la ecuación (2.15).

$$\nabla J(\boldsymbol{\theta}) \propto \sum_{s\in\mathcal{S}} \mu_{\pi_{\boldsymbol{\theta}}}(s) \sum_{a\in\mathcal{A}} (Q_{\pi_{\boldsymbol{\theta}}}(s,a) - b(s)) \cdot \nabla \pi_{\boldsymbol{\theta}}(s,a). \tag{2.17}$$

De esta manera, se consigue reducir la varianza pero no se introduce un cambio en el gradiente, pues no depende de las acciones:

$$\sum_{a\in\mathcal{A}} b(s) \cdot \nabla \pi_{\boldsymbol{\theta}}(s,a) = b(s) \sum_{a\in\mathcal{A}} \nabla \pi_{\boldsymbol{\theta}}(s,a) = b(s) \cdot \nabla \sum_{a\in\mathcal{A}} \pi_{\boldsymbol{\theta}}(s,a) = b(s) \cdot \nabla 1 = 0.$$

En definitiva, la regla de actualización de los parámetros en el algoritmo REINFORCE con *baseline* es:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \cdot \left(\sum_{k=t}^{T} \gamma^{k-t} \cdot r(s_k, a_k, s_{k+1}) - b(s_t) \right) \cdot \nabla \log \left(\pi_{\boldsymbol{\theta}_t}(s_t, a_t) \right). \tag{2.18}$$

Observación 2.1.7. *Una elección natural del baseline es utilizar una estimación de la función de valor, $b(S_t) \coloneqq \hat{V}(S_t)$, por lo que, en muchas ocasiones, los métodos basados en el valor complementan a los métodos basados en la política, y viceversa.*

Métodos actor-crítico

La principal restricción del algoritmo REINFORCE, y de métodos análogos, es que se debe realizar el cálculo completo sobre los T pasos antes de actualizar los parámetros. Esto no permite una actualización *online* de los parámetros, ralentizando el entrenamiento.

Para sobrevenir este problema se utiliza una adaptación del algoritmo *Q-learning* (2.10), utilizando una estimación del *baseline* $b(s_t) \coloneqq \hat{V}(s_t)$ no solo para el primer paso de la transición, sino también para el segundo: $\hat{V}(s_{t+1})$. Así, la estimación se obtiene en base a la relación:

$$\hat{V}(s_t) \leftarrow \hat{V}(s_t) + \alpha \left[r(s_t, a_t) + \gamma \cdot \hat{V}(s_{t+1}) - \hat{V}(s_t) \right]. \tag{2.19}$$

De esta forma aparecen los métodos actor-crítico [4], o *actor-critic* en inglés, que combinan el aprendizaje de la función de valor con la actualización de los parámetros de la política. El *actor* actualiza la política utilizando la información aportada por el *crítico*, que estima la función de valor $\hat{V}(s)$.

Así, reemplazando el *baseline* de la ecuación (2.18), utilizando un único paso ($T = t+1$) y estimando el resto:

$$\sum_{k=t}^{T} \gamma^{k-t} \cdot r(s_k, a_k, s_{k+1}) = r(s_t, a_t, s_{t+1}) + \sum_{k=t+1}^{T} \gamma^{k-t} \cdot r(s_k, a_k, s_{k+1}) =$$
$$= r(s_t, a_t, s_{t+1}) + \gamma \cdot \hat{V}(s_{t+1}).$$

En definitiva, la actualización de los parámetros de la política en el método actor-crítico es:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \cdot \Big(r(s_t, a_t, s_{t+1}) + \gamma \cdot \hat{V}(s_{t+1}) - \hat{V}(s_t)\Big) \cdot \nabla \log\left(\pi_{\boldsymbol{\theta}_t}(s_t, a_t)\right). \quad (2.20)$$

Observación 2.1.8. *Como se puede observar en la ecuación* (2.20)*, el método de actor-crítico utiliza una adaptación del TD-error (término entre corchetes) para su actualización. Además, se puede generalizar al caso continuo $T \to \infty$ de manera sencilla.*

Algoritmo *Proximal Policy Optimization* (PPO)

Uno de los avances más relevantes en los métodos actor-crítico ha sido el desarrollo del algoritmo *Proximal Policy Optimization* (PPO), propuesto por Schulman et al. [54]. Este algoritmo busca optimizar políticas estocásticas maximizando una función objetivo basada en el gradiente de política, al tiempo que restringe los cambios drásticos en los parámetros de la política, lo que contribuye a mejorar la estabilidad y la eficiencia del entrenamiento.

El punto de partida del algoritmo PPO es considerar el ratio de probabilidades de la nueva política con respecto a la antigua:

$$r_t(\boldsymbol{\theta}) = \frac{\pi_{\boldsymbol{\theta}}(s_t, a_t)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(s_t, a_t)}. \quad (2.21)$$

Este ratio mide cuánto cambia la probabilidad de seleccionar una acción dada bajo la nueva política en comparación con la antigua. Utilizando este ratio, se define la siguiente función objetivo a maximizar:

$$\mathcal{L}^{\text{PPO}}(\boldsymbol{\theta}) = \mathbb{E}_t\left[\min\left(r_t(\boldsymbol{\theta}) \cdot A_t,\ \text{clip}(r_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon) \cdot A_t\right)\right], \quad (2.22)$$

donde A_t es una estimación del valor ventaja, y $\epsilon > 0$ es un hiperparámetro que determina cuánto se permite variar la política entre iteraciones. El término *clip* actúa como un regulador que impide que el ratio $r_t(\boldsymbol{\theta})$ se aleje demasiado de 1. Si el cambio propuesto es muy grande, la función objetivo es truncada, lo que penaliza actualizaciones que podrían desestabilizar el entrenamiento.

Además, la estimación del valor ventaja A_t suele obtenerse mediante métodos como el *Generalized Advantage Estimation* (GAE) [55]. Este método propone una interpolación entre dos extremos comunes: la estimación Monte Carlo completa (alta varianza, bajo sesgo) y la estimación basada en un solo paso de *bootstrapping* (baja varianza, alto sesgo). Para ello, GAE introduce un parámetro $\lambda \in [0, 1]$ que controla este compromiso sesgo-varianza, definiendo la ventaja generalizada como:

$$A_t^{\text{GAE}(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \cdot \delta_{t+l}, \text{ con } \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t). \quad (2.23)$$

Este cálculo puede interpretarse como una suma ponderada exponencialmente de los errores temporales futuros, donde los pesos dependen de γ (factor de descuento) y λ (factor de suavizado). Valores de λ cercanos a 1 incorporan más pasos futuros en la estimación, mientras que valores bajos de λ dan lugar a estimaciones más locales.

2.2. Aprendizaje por refuerzo multiagente

El aprendizaje por refuerzo en entornos multiagente no es una generalización trivial del aprendizaje por refuerzo de un único agente. De hecho, la estructura del entorno y el objetivo del juego determinan la forma de aprendizaje, además de imponer restricciones y dificultades adicionales. Aun así, el esquema es análogo al de un único agente: cada agente toma decisiones eligiendo acciones, y a cambio recibe información sobre el nuevo estado del entorno y una señal de recompensa, que refleja si su acción ha sido beneficiosa o no con respecto a un objetivo determinado (véase la figura 2.4).

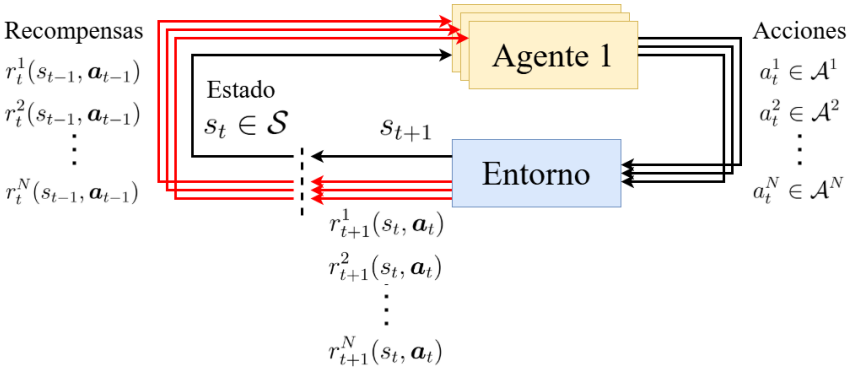


Figura 2.4: *Esquema general del aprendizaje por refuerzo en un entorno multiagente. El agente i observa el estado del entorno $s_t \in \mathcal{S}$, selecciona una acción $a_t^i \in \mathcal{S}$, recibe una recompensa $r_{t+1}^i \in \mathbb{R}$ y una nueva observación del estado $s_{t+1} \in \mathcal{S}$.*

La diferencia principal en este tipo de entornos es que el estado al que se transita no depende exclusivamente de las acciones del agente, sino de las acciones de todos los agentes que participan. Esto lleva a un problema más complejo, no estacionario y con distintos algoritmos [11].

Los entornos de aprendizaje multiagente se estudian mediante los juegos de Markov, *Markov games* en inglés, que se definen como sigue [57].

Definición 2.2.1. *Un juego de Markov (Markov game) se define como una tupla $(N, \mathcal{S}, \mathcal{A}^1, \ldots, \mathcal{A}^N, p, r^1, \ldots, r^N)$, donde:*

- *N es el número de agentes, o jugadores.*
- *$\mathcal{S}$ es el conjunto de estados.*
- *$\mathcal{A}^i$ es el conjunto de acciones del agente $i \in \{1, \ldots, N\}$.*
- *$p(s'|s, \boldsymbol{a})$ es la función de transición, que indica la probabilidad de que el entorno pase del estado s al estado s' al ejecutarse la acción $\boldsymbol{a} := (a^1, \ldots, a^N)$, denominada acción conjunta. Se cumple entonces que $\sum_{s' \in \mathcal{S}} p(s'|s, a) = 1$, $\forall s \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}$, donde la acción conjunta pertenece a $\mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N$.*
- *$r^i(s, \boldsymbol{a}, s')$ es la función que representa la recompensa instantánea esperada que el agente i obtiene al realizarse la acción conjunta $\boldsymbol{a}$ en el estado s, pasando al estado s'.*

Observación 2.2.2. *La principal diferencia entre la definición 2.2.1 de los juegos de Markov y la definición 2.1.1 de los Procesos de Decisión de Markov es que tanto la función de transición como las funciones de recompensa dependen de la acción conjunta, i.e. de la acción de todos los agentes involucrados.*

Por supuesto, la noción de política también se extiende a los juegos de Markov, aunque individualizada sobre cada agente. Esto es, cada agente i tiene una política $\pi^i : \mathcal{S} \times \mathcal{A}^i \to [0, 1]$, entendida como una distribución de probabilidad sobre el espacio de acciones del agente y condicionada al estado dado.

En el caso de los juegos de Markov se define también una "política conjunta" [57], o *joint policy* en inglés, $\boldsymbol{\pi} \coloneqq \pi^1 \times \cdots \times \pi^N$, que se puede entender como un perfil de política que seguiría un agente omnipotente, i.e. un agente controlando a todos los agentes. Esta política conjunta satisface que $\boldsymbol{\pi} : \mathcal{S} \times \mathcal{A} \to [0, 1]$, donde $\boldsymbol{a} = (a^1, \ldots, a^N) \in \mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N$. Resulta necesario definir además la política conjunta que excluye a un agente dado i, de la forma $\boldsymbol{\pi}^{-i} \coloneqq \prod_{j \neq i} \pi^j$, siendo que $\boldsymbol{\pi} = \pi^i \times \boldsymbol{\pi}^{-i}$, denotado como $\boldsymbol{\pi} \coloneqq \langle \pi^i, \boldsymbol{\pi}^{-i} \rangle$.

Con esta extensión del concepto de política, resulta interesante notar que la función de valor será distinta para cada agente, pues las recompensas asociadas son distintas para cada agente. Aun así, esta función depende de la política conjunta, y no de las políticas individuales de los agentes. La función de valor del agente i es:

$$V^i_{\boldsymbol{\pi}}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r^i(S_t, \boldsymbol{A}_t, S_{t+1}) \mid \boldsymbol{A}_t \sim \boldsymbol{\pi}(\cdot \mid S_t),\ S_0 = s\right], \tag{2.24}$$

y la función de acción-valor partiendo del estado s y la acción conjunta $\boldsymbol{a}$ es:

$$Q^i_{\boldsymbol{\pi}}(s, \boldsymbol{a}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r^i(S_t, \boldsymbol{A}_t, S_{t+1}) \mid \boldsymbol{A}_t \sim \boldsymbol{\pi}(\cdot \mid S_t),\ S_0 = s, \boldsymbol{A}_0 = \boldsymbol{a}\right]. \tag{2.25}$$

De manera clara, la función de acción-valor también es diferente para cada agente, pero depende de la acción conjunta y no de la acción individual del agente. Por ende, los conceptos de optimalidad individual y global van a diferir, pues carece de sentido definir la calidad de una política individual sin relacionarla con las políticas del resto de agentes.

Se define entonces la "mejor respuesta" [43], o *best response* en inglés, con respecto a la política conjunta $\boldsymbol{\pi}^{-i}$, como la política $\pi^{i,*}$ que satisface que:

$$\forall \pi^i,\ \forall s \in \mathcal{S} \quad V^i_{\langle \pi^i, \boldsymbol{\pi}^{-i} \rangle}(s) \leq V^i_{\langle \pi^{i,*}, \boldsymbol{\pi}^{-i} \rangle}(s). \tag{2.26}$$

Observación 2.2.3. *En general, la mejor respuesta no tiene por qué ser una única política. Por ello, se define el conjunto de las políticas que son mejor respuesta para $\boldsymbol{\pi}^{-i}$ como $BR(\boldsymbol{\pi}^{-i})$.*

De esta manera, la mejor respuesta es el análogo al concepto intuitivo de política óptima individual, pues, fijando el resto de políticas, es la que maximiza la función de valor del agente. Así, la definición de "equilibrio de Nash" es directa [43].

Definición 2.2.4. *Dado un juego de Markov con N agentes, si la política conjunta $\boldsymbol{\pi} = \pi^1 \times \cdots \times \pi^N$ satisface que, $\forall i \in \{1, \ldots, N\}$, $\pi^i \in BR(\boldsymbol{\pi}^{-i})$, entonces $\boldsymbol{\pi}$ es un equilibrio de Nash.*

Observación 2.2.5. *De la definición 2.2.4 se puede deducir que el equilibrio de Nash asume que las políticas de cada agente son independientes, i.e. las acciones de los agentes no están correlacionadas [73].*

2.2.1. Observación parcial en juegos de Markov

Una característica clave de los juegos de Markov es su posible extensión a juegos parcialmente observables. Esto es, se puede considerar que algunos agentes, o incluso todos, reciben información parcial del estado, i.e. "observan" solo una parte del entorno. En este contexto, cuando los agentes no tienen acceso completo al estado del entorno, se habla de juegos de Markov parcialmente observables (POMG, del inglés *Partially Observable Markov Games*). Esta extensión es una generalización natural que se puede aplicar tanto a los juegos de Markov como a los procesos de decisión parcialmente observables (POMDP), introduciendo un mayor realismo en entornos donde los agentes no pueden observar directamente el estado global del sistema [17].

Definición 2.2.6. *Un juego de Markov parcialmente observable (POMG) se define como una tupla $(N, \mathcal{S}, \mathcal{A}^1, \ldots, \mathcal{A}^N, p, O, r^1, \ldots, r^N)$ donde:*

- *N, $\mathcal{S}$, $\mathcal{A}^i$, p, y r^i son los de la definición de juego de Markov (definición 2.2.1).*
- *$O(s, i)$ es la función de observación que permite obtener el espacio de observaciones del agente i, dado por: $\mathcal{O}^i = \{o^i | s \in \mathcal{S}, o^i = O(s, i)\}$.*

En este caso, las políticas de los agentes ya no pueden estar directamente condicionadas al estado s, pues no es completamente observable. En su lugar, cada política se define como $\pi^i : \mathcal{O}^i \times \mathcal{A}^i \to [0, 1]$, donde $\mathcal{O}^i$ es el conjunto de observaciones del agente i [46].

Desde un punto de vista cualitativo, los POMG modelan situaciones donde la información es limitada, ruidosa o local. La falta de observación completa implica que los agentes deben enfrentarse a una incertidumbre adicional, no solo sobre el resultado de sus acciones, sino también sobre el estado actual del entorno y sobre las decisiones del resto de agentes. Así, se desarrollan estrategias más complejas en las que puede ser necesario mantener memorias internas, construir creencias (distribuciones de probabilidad sobre los estados posibles) o utilizar el histórico de acciones y observaciones para el aprendizaje [19, 22].

En general, la observación parcial es especialmente relevante en entornos reales, como los juegos de cartas (cada jugador tiene una visión parcial del estado, las cartas en su mano y las jugadas anteriores, y debe inferir información sobre las cartas ocultas y la estrategia de los oponentes [7]), la gestión autónoma del tráfico (cada vehículo percibe solo una parte del entorno, como los vehículos cercanos y las señales de tráfico visibles, y debe tomar decisiones cooperativas sin conocer completamente la intención de los demás [56]) o los videojuegos multijugador en tiempo real (los

jugadores tienen una visión limitada del mapa, y están obligados a anticiparse a los movimientos desconocidos del adversario [63]).

Observación 2.2.7. *Todo el marco teórico desarrollado para juegos de Markov completamente observables —incluyendo políticas, funciones de valor, mejores respuestas y equilibrios de Nash— se puede extender formalmente al caso parcialmente observable [46]. No obstante, debido al incremento en la complejidad matemática y a fin de mantener la claridad y brevedad de este trabajo, en lo que sigue se tratará exclusivamente el caso completamente observable, refiriéndose a este simplemente como "juegos de Markov".*

2.2.2. Configuraciones en entornos multiagente

Dada la descripción de los entornos multiagente como juegos de Markov, se pueden distinguir tres configuraciones con distintos objetivos: cooperativos, competitivos y mixtos (véase la figura 2.5).

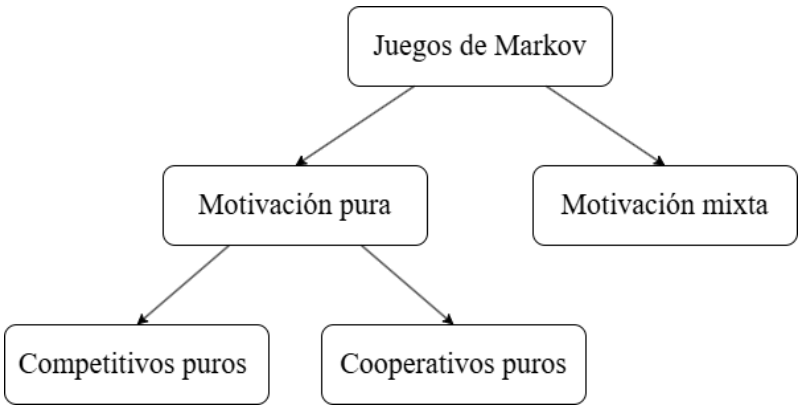


Figura 2.5: *Taxonomía de entornos multiagente modelados como juegos de Markov, clasificados según el tipo de interacción entre agentes. Representación inspirada en [53].*

En general, un juego "competitivo" se representa por un juego de Markov de suma cero, i.e. un juego en el que $\sum_{i=1}^{N} r^i(s, \boldsymbol{a}, s') = 0$ para cualesquiera $(s, \boldsymbol{a}, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ [57]. De esta forma, y en el caso más común que es el de dos agentes, la ganancia de uno conlleva directamente la pérdida del otro.

Por otro lado, se considera que un juego "cooperativo" es aquel en el que la función de recompensa es única [9], es decir, en el que $r^1 = \cdots = r^N =: r$. En esta situación, tanto la función de valor como la función de acción-valor son idénticas para todos los agentes, pudiendo emplearse las técnicas de aprendizaje de los entornos de un solo agente.

Además, se puede considerar una variación de la cooperación pura en la que los agentes poseen funciones de recompensa distintas, pero con el objetivo de maximizar la recompensa promedio $\bar{r} = \frac{1}{N} \sum_{i=1}^{N} r^i(s, \boldsymbol{a}, s')$, para cualesquiera $(s, \boldsymbol{a}, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$. En este contexto, las técnicas de aprendizaje de un solo agente no pueden aplicarse, pues las funciones de valor y de acción-valor serán distintas para cada agente, al depender de la recompensa asociada a cada uno. Por ello, se deben definir dinámicas nuevas que permitan la optimización de las políticas individuales de los agentes.

Por último, se pueden encontrar situaciones intermedias conocidas como juegos de "motivación mixta", en los que los agentes persiguen objetivos parcialmente alineados pero no completamente coincidentes [32]. En estos juegos, cada agente posee su propia función de recompensa r^i, que puede coincidir parcialmente con la de los demás agentes, pero no necesariamente en todo momento ni en todos los estados. En términos formales, los juegos de motivación mixta no son ni de suma cero ni puramente cooperativos, por lo que se denominan "de suma general" (*general-sum* en inglés) y requieren que los agentes aprendan a equilibrar el beneficio individual con el impacto de sus acciones sobre los demás.

2.2.3. Dinámicas de entrenamiento en entorno multiagente

En el marco descrito anteriormente, i.e. en juegos cooperativos con recompensas individuales distintas, aunque los agentes buscan colectivamente maximizar una recompensa promedio, las diferencias en las funciones de recompensa individuales impiden una aplicación directa de enfoques tradicionales de aprendizaje por refuerzo de agente de un solo agente. Para abordar esta complejidad, es necesario definir nuevas dinámicas de entrenamiento que permitan optimizar las políticas individuales de forma coherente con los objetivos colectivos.

En el aprendizaje por refuerzo multiagente (MARL), el proceso de entrenamiento consiste en optimizar las políticas a partir de la experiencia adquirida (estados, acciones, recompensas, etc.), mientras que la ejecución implica que los agentes interactúan con el entorno tomando decisiones de acuerdo con políticas individuales o conjuntas. Dependiendo de si los agentes requieren información de los demás durante la actualización de sus políticas, se puede distinguir entre *entrenamiento centralizado* y *entrenamiento descentralizado*. De manera análoga, según si se necesita información externa en tiempo de ejecución, se diferencia entre *ejecución centralizada* y *ejecución descentralizada*.

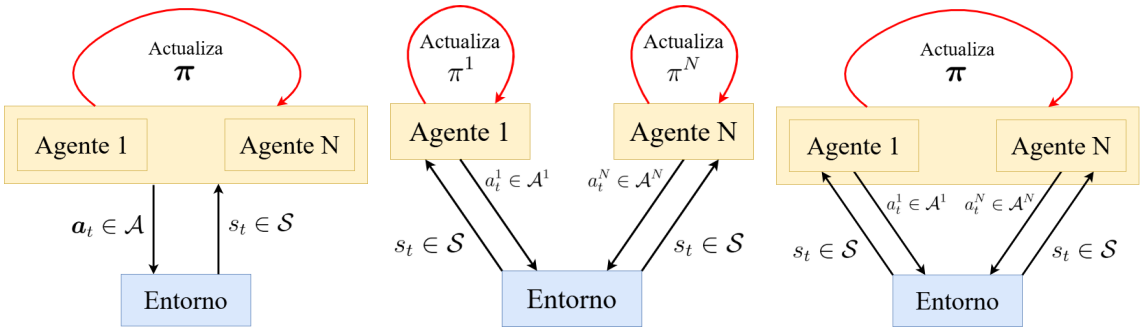
Combinando estas dimensiones, surgen tres paradigmas fundamentales en MARL (véase la figura 2.6):

a) Entrenamiento centralizado con ejecución centralizada (CTCE, por sus siglas en inglés *Centralized Training with Centralized Execution*): cada agente actualiza y ejecuta su política utilizando la información global, i.e. intercambiando información con el resto de agentes. Esto es, se actualiza y ejecuta la política global $\boldsymbol{\pi} : \mathcal{S} \times \mathcal{A} \to [0, 1]$.

b) Entrenamiento descentralizado con ejecución descentralizada (DTDE, por sus siglas en inglés *Decentralized Training with Decentralized Execution*): cada agente actualiza su política independientemente y la ejecuta utilizando únicamente su información local, sin intercambio de información. Esto es, se actualiza y ejecuta la política individual $\pi^i : \mathcal{S} \times \mathcal{A}^i \to [0, 1]$.

c) Entrenamiento centralizado con ejecución descentralizada (CTDE, por sus siglas en inglés *Centralized training with decentralized execution*): cada agente actualiza su política utilizando de la información global pero ejecuta su política individual utilizando únicamente su información local, sin intercambio de

información. Esto es, se actualiza la política $\boldsymbol{\pi} = \pi^1 \times \cdots \times \pi^N : \mathcal{S} \times \mathcal{A} \to [0, 1]$, pero cada agente ejecuta la política individual $\pi^i : \mathcal{S} \times \mathcal{A}^i \to [0, 1]$.



(a) *CTCE: Entrenamiento centralizado y ejecución centralizada.*

(b) *DTDE: Entrenamiento descentralizado y ejecución descentralizada.*

(c) *CTDE: Entrenamiento centralizado y ejecución descentralizada.*

Figura 2.6: *Paradigmas fundamentales en aprendizaje por refuerzo multiagente (MARL).*

Este último resulta especialmente atractivo, ya que permite aprovechar información global y compartida (como estados completos o acciones de todos los agentes) durante el entrenamiento, mientras que en la fase de ejecución cada agente actúa únicamente con su observación local, respetando así las restricciones de descentralización propias de muchos entornos reales.

CTCE - *Métodos actor-crítico*

Como se puede comprobar por su definición, el CTCE permite de manera natural emplear los algoritmos clásicos del aprendizaje por refuerzo de un solo agente, sin más que considerar las acciones y políticas conjuntas, i.e. $\boldsymbol{a} = (a^1, \ldots, a^N)$ y $\boldsymbol{\pi} = \pi^1 \times \cdots \times \pi^N$, y la recompensa promedio $\bar{r}$, y se pueden definir funciones conjuntas de valor:

$$\boldsymbol{V}_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot \bar{r}(S_t, \boldsymbol{A}_t, S_{t+1}) \mid \boldsymbol{A}_t \sim \boldsymbol{\pi}(\cdot \mid S_t),\ S_0 = s\right], \tag{2.27}$$

y de acción-valor:

$$\boldsymbol{Q}_\pi(s, \boldsymbol{a}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot \bar{r}(S_t, \boldsymbol{A}_t, S_{t+1}) \mid \boldsymbol{A}_t \sim \boldsymbol{\pi}(\cdot \mid S_t),\ S_0 = s, \boldsymbol{A}_0 = \boldsymbol{a}\right], \tag{2.28}$$

que se busca maximizar, aplicando los algoritmos clásicos de un solo agente presentados en la sección 2.1, como es el caso de los métodos actor-crítico descritos en la sección 2.1.2.

DTDE - *Métodos actor-crítico independientes*

Como ocurría para los algoritmos de un solo agente, estos pueden dividirse en algoritmos basados en la política y algoritmos basados en el valor. Además, en este

caso, dado que se está considerando un entrenamiento descentralizado, se dice que los agentes son "aprendices independientes" (*independent learners*), en contraposición a los "aprendices de acción conjunta" (*joint-action learners*) [9]. Por ello, se definen funciones de acción-valor independientes:

$$Q^i_{\pi^i}(s, a^i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot \bar{r}(S_t, A^i_t, S_{t+1}) \mid \boldsymbol{A}^i_t \sim \pi^i(\cdot \mid S_t),\ S_0 = s, \boldsymbol{A}^i_0 = a^i\right].$$

que se optimizan iterativamente utilizando los estados anterior y posterior, así como la recompensa promedio $\bar{r}$ asociada a la acción tomada por el agente, junto con las acciones tomadas por el resto de agentes. Esto es, el algoritmo *Q-learning* independiente (IQL, del inglés *Independent Q-learning*) es:

$$Q^i_{\pi^i}(s_t, a^i_t) \leftarrow Q^i_{\pi^i}(s_t, a^i_t) + \alpha\left[\bar{r}(s_t, a^i_t, s_{t+1}) + \gamma \cdot \max_{a^i} Q(s_{t+1}, a^i) - Q(s_t, a^i_t)\right]. \tag{2.29}$$

La principal debilidad de este paradigma de entrenamiento es ya apreciable: el resto de agentes son tratados como parte del entorno, provocando que este sea no estacionario y las transiciones $(s_t, a^i_t) \to s_{t+1}$ no sean concretas, pues en realidad se transita $(s_t, a^i_t, a^{-i}_t) \to s_{t+1}$. De hecho, la función $\bar{r}$ no está bien definida de esta forma, pues en la práctica depende de la acción conjunta $\boldsymbol{a}$ y no de la acción individual a^i.

En definitiva, como en general el resto de agentes también está entrenando, i.e. actualizando sus políticas, el algoritmo IQL puede no converger [61], y las condiciones bajo las que lo hace siguen siendo un área de estudio [21].

Observación 2.2.8. *Para resolver el problema de la no estacionariedad se pueden mantener fijas las políticas del resto de agentes, y entrenar al agente con métodos de aprendizaje para un solo agente. Esto resulta en un mínimo local y en la obtención de la "mejor respuesta" para el agente i [3]. Aun así, entrenar de esta forma requiere coordinación para comunicar qué agente está aprendiendo en cada instante, saliéndose del paradigma DTDE.*

Por otro lado, los métodos actor-crítico, originalmente desarrollados para agentes individuales, pueden extenderse de manera natural al entorno multiagente con entrenamiento descentralizado, permitiendo que cada agente aprenda de forma independiente. En este contexto, se asume que cada agente i sigue una política $\pi^i_{\boldsymbol{\theta}^i}(s, a^i)$, donde $\boldsymbol{\theta}^i \in \mathbb{R}^{d^i}$ son los parámetros de la política, a^i es la acción y s es el estado observado por el agente.

La generalización natural a los métodos actor-crítico independientes (IAC, siglas en inglés de *Independent actor-critic*) se obtiene como una combinación del algoritmo *Q-learning* independiente (IQL) y la descentralización del algoritmo REINFORCE [16]. Cada agente consta de un crítico, que estima una función de valor $\hat{V}^i(s)$, y un actor, que actualiza una política $\pi^i_{\boldsymbol{\theta}^i}$ utilizando la estimación obtenida del crítico.

En definitiva, la actualización del crítico se realiza adaptando el algoritmo IQL:

$$\hat{V}^i(s_t) \leftarrow \hat{V}^i(s_t) + \alpha\left[\bar{r}(s_t, a^i_t) + \gamma \cdot \hat{V}^i(s_{t+1}) - \hat{V}^i(s_t)\right], \tag{2.30}$$

y la actualización de los parámetros de la política, realizada por el actor, es:

$$\boldsymbol{\theta}_{t+1}^{i} = \boldsymbol{\theta}_{t}^{i} + \alpha \cdot \left(\bar{r}(s_t, a_t^i, s_{t+1}) + \gamma \cdot \hat{V}^i(s_{t+1}) - \hat{V}^i(s_t)\right) \cdot \nabla \log \left(\pi_{\boldsymbol{\theta}_t^i}^{i}(s_t, a_t^i)\right). \quad (2.31)$$

Observación 2.2.9. *Al aplicarse en el paradigma DTDE, los métodos actor-crítico independientes, y en general los métodos basados en la política, ofrecen garantías más sólidas de convergencia local en comparación con los métodos basados en el valor [75, 76].*

CTDE - *Métodos actor-crítico con crítico centralizado*

Con el objetivo de solventar las carencias claras del paradigma DTDE —no estacionariedad y reducida convergencia— y desarrollar técnicas de entrenamiento aplicables en casos reales, se considera un paradigma híbrido: CTDE. Este paradigma ha permitido desarrollar métodos particularmente eficaces en entornos multiagente, ya que durante la fase de entrenamiento los agentes pueden aprovechar información global, como los estados y acciones de todos los agentes, que luego no estará disponible durante la ejecución. Así, la ejecución es descentralizada, luego viable a nivel práctico. De nuevo, las arquitecturas en este paradigma pueden agruparse en las de aprendizaje basadas en el valor y las basadas en la política.

Dentro de las primeras, las *Value Decomposition Networks* (VDN) [59] marcaron un punto de partida en los métodos de factorización de funciones de valor dentro del aprendizaje por refuerzo profundo multiagente. Su principal aporte radica en la descomposición de la función Q conjunta en una suma de funciones Q^i individuales, una por cada agente. Esta idea, aunque conceptualmente sencilla, permite que el entrenamiento se lleve a cabo de forma centralizada, mientras que la ejecución puede mantenerse descentralizada, ya que cada agente toma decisiones basadas únicamente en su propia función Q^i. La hipótesis se expresa como:

$$Q(s, \boldsymbol{a}) \approx \sum_{i=1}^{N} Q^i(s, a^i). \quad (2.32)$$

De esta manera, cada agente utiliza una red neuronal para calcular su función Q^i, a partir de sus observaciones locales. Durante el entrenamiento, las funciones Q^i se suman para formar una estimación de la acción-valor conjunta, lo que permite aplicar una función de pérdida similar a la utilizada en DQN de un solo agente (véase la sección 2.1.1). Una vez finalizado el entrenamiento, cada agente conserva su red individual y puede seleccionar acciones maximizando su función Q^i:

$$\pi^i(s) = \arg\max_{a^i} Q^i(s, a^i). \quad (2.33)$$

Observación 2.2.10. *Aunque la aproximación solo es exacta en escenarios de independencia total entre agentes, en muchos casos sigue permitiendo tomar decisiones efectivas.*

Siguiendo esta línea de desarrollo, QMIX [48] permite una factorización más general de la función Q conjunta. En lugar de asumir que esta se puede expresar

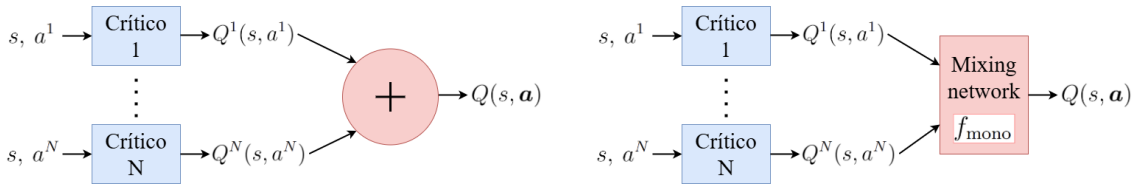
como una simple suma de funciones Q^i, como en VDN, QMIX postula que la función Q conjunta es una función monótona de las funciones Q^i individuales. Es decir, un aumento en el valor de cualquier Q^i implica un aumento (o al menos no una disminución) en el valor conjunto.

Formalmente, QMIX plantea que la función Q conjunta se puede aproximar como:

$$Q(s, \boldsymbol{a}) \approx f_{\text{mono}}(Q^1(s, a^1), \ldots, Q^N(s, a^N)), \tag{2.34}$$

donde f_{mono} es una función monótona aprendida durante el entrenamiento. Esta propiedad asegura que la acción conjunta que maximiza la función Q también puede encontrarse seleccionando las acciones que maximizan cada función Q^i individual, lo cual mantiene la ejecución descentralizada al igual que VDN, pero permite una mayor flexibilidad funcional.

De manera análoga a VDN, la arquitectura de QMIX utiliza redes neuronales para cada agente, que procesan la observación del agente y generan los valores Q^i para todas sus posibles acciones. En el caso QMIX, los valores se introducen en una red de combinación (*mixing network* en inglés) que produce el valor Q, estando diseñada para ser monótona mediante la restricción de pesos no negativos. Pueden observarse las diferencias estructurales entre QMIX y VDN en la figura 2.7.



(a) *VDN* (b) *QMIX*

Figura 2.7: *Comparativa de los métodos CTDE enfocados en la función de acción valor.*

Observación 2.2.11. *A pesar de sus ventajas, QMIX también tiene limitaciones: su capacidad para aproximar correctamente la función Q conjunta depende de que las decisiones de los agentes no estén fuertemente interrelacionadas.*

Por otro lado, una de las estrategias más exitosas dentro de este paradigma, pero en el contexto del aprendizaje basado en la política, consiste en emplear un esquema *actor-crítico* con un crítico centralizado, destacando algoritmos como MADDPG [34] y COMA [16].

Una versión básica de esta estrategia es el método MADDPG (del inglés *Multi-Agent Deep Deterministic Policy Gradient*) [34], donde cada actor descentralizado se entrena usando un único crítico centralizado que estima la función de valor conjunta $\hat{V}(s)$. Este crítico centralizado permite evaluar el impacto de la acción del agente i, pero en el contexto de las acciones del resto de agentes. Así, se actualiza de manera análoga a la función de valor de un solo agente:

$$\hat{V}(s_t) \leftarrow \hat{V}(s_t) + \alpha \left[r(s_t, \boldsymbol{a}_t) + \gamma \cdot \hat{V}(s_{t+1}) - \hat{V}(s_t) \right]. \tag{2.35}$$

Con dicha estimación, cada actor actualiza los parámetros $\boldsymbol{\theta}^i$ de su política individual $\pi^i_{\boldsymbol{\theta}^i}$, utilizando el mismo principio que se empleaba en el IAC:

$$\boldsymbol{\theta}^i_{t+1} = \boldsymbol{\theta}^i_t + \alpha \cdot \left(\bar{r}(s_t, a^i_t, s_{t+1}) + \gamma \cdot \hat{V}(s_{t+1}) - \hat{V}(s_t) \right) \cdot \nabla \log \left(\pi^i_{\boldsymbol{\theta}^i_t}(s_t, a^i_t) \right). \tag{2.36}$$

Observación 2.2.12. *Cabe destacar que el crítico, aunque es centralizado, evalúa políticas descentralizadas, lo cual es correcto en el marco CTDE. Además, bajo supuestos razonables, MADDPG converge a un óptimo local [36].*

Tras introducir el enfoque de MADDPG, el algoritmo COMA (del inglés *Counterfactual Multi-Agent Policy Gradients*) [16] introduce una mejora significativa al tratamiento de la asignación de crédito entre agentes mediante un *baseline* contrafactual. A diferencia del uso estándar de una función de ventaja $A(s, \boldsymbol{a}) = Q(s, \boldsymbol{a}) - V(s)$, COMA calcula una ventaja específica por agente restando la contribución esperada de dicho agente al valor conjunto. Esto es, la ventaja contrafactual de un agente i se define como:

$$A^i(s, \boldsymbol{a}) = Q(s, \boldsymbol{a}) - \sum_{a^i} \pi^i(s, a^i) \cdot Q(s, (a^i, \boldsymbol{a}^{-i})) \tag{2.37}$$

Este término mide cuánto mejor (o peor) fue la acción real del agente comparada con una expectativa ponderada sobre sus posibles acciones, manteniendo las acciones del resto constantes. Así, COMA utiliza un crítico centralizado que genera directamente $Q(s, (a^i, \boldsymbol{a}^{-i}))$ y, por ende, el valor de ventaja A^i. Con ello, la política se actualiza de manera similar a MADDPG, pero usando la ventaja específica del agente en lugar de la función de valor.

Una evolución destacada dentro de esta línea es MAPPO (del inglés *Multi-Agent Proximal Policy Optimization*) [71], que combina la robustez del algoritmo PPO [54] (véase la ecuación (2.22)) con el paradigma CTDE. En MAPPO, cada agente posee una política propia que se actualiza de forma descentralizada, mientras que el crítico centralizado aprovecha las observaciones y acciones de todos los agentes durante el entrenamiento. La clave del enfoque radica en aplicar el principio de optimización proximal (PPO) en un entorno multiagente, lo que permite una mejora estable y eficiente de las políticas individuales sin sobrepasar los límites de confianza establecidos.

Durante el entrenamiento, el crítico centralizado estima ventajas generalizadas (GAE, por sus siglas en inglés), mientras que los actores actualizan sus políticas maximizando una función objetivo penalizada, que evita grandes desviaciones respecto a la política anterior:

$$\mathcal{L}^{\text{MAPPO}}(\boldsymbol{\theta}^i) = \mathbb{E}\left[\text{mín}\left(r(\boldsymbol{\theta}^i) \cdot A^i, \text{clip}(r(\boldsymbol{\theta}^i), 1-\epsilon, 1+\epsilon) \cdot A^i\right)\right], \tag{2.38}$$

donde $r(\boldsymbol{\theta}^i) = \frac{\pi_{\boldsymbol{\theta}^i}(s,a^i)}{\pi_{\boldsymbol{\theta}^i_{\text{old}}}(s,a^i)}$ y A^i es la ventaja estimada por la ecuación (2.37). Este enfoque simplemente trata de maximizar la mejora de la política nueva con respecto a la antigua, y ha demostrado una gran estabilidad y rendimiento competitivo en múltiples entornos cooperativos, como los propuestos por el *benchmark* SMAC [71].

CAPÍTULO 3

Teoría de juegos y dilemas sociales

La teoría de juegos proporciona un marco formal eficaz para analizar interacciones estratégicas entre agentes racionales, especialmente en entornos multiagente donde los intereses individuales pueden entrar en conflicto con el bienestar colectivo. Dentro de este marco, los dilemas sociales (*social dilemmas* en inglés) constituyen una clase fundamental de problemas que capturan este conflicto: situaciones en las que actuar de manera egoísta proporciona beneficios inmediatos al individuo, pero si todos los agentes adoptan dicha conducta, el resultado global es subóptimo.

Estas situaciones, formalizadas mediante juegos de motivación mixta (o suma no nula), permiten estudiar con precisión la tensión entre la racionalidad individual y la colectiva [28]. La cooperación puede generar resultados más beneficiosos para todos los individuos que los que podrían alcanzar de forma aislada, pero existen fuertes incentivos a la deserción o al aprovechamiento gratuito del esfuerzo ajeno. Este fenómeno es emblemático en la *tragedia de los comunes* [18], donde el uso egoísta de recursos compartidos puede conducir al colapso colectivo.

Desde el punto de vista del aprendizaje por refuerzo, los dilemas sociales plantean un reto central: las políticas óptimas individuales no coinciden con las políticas que maximizarían el bienestar colectivo. Comprender cómo emergen o colapsan dinámicas cooperativas en estos entornos es crucial para diseñar agentes autónomos que interactúen de manera robusta y alineada en sistemas complejos.

3.1. Dilemas sociales clásicos y juegos de matrices

Los dilemas sociales se han estudiado tradicionalmente en contextos estáticos y simplificados, donde las decisiones se toman simultáneamente en un único paso. Es-

tos entornos se representan mediante juegos de matrices (del inglés *matrix games*), que son juegos de diferente motivación (pura o mixta) entre varios jugadores definidos por matrices, denominadas "matrices de pagos" [65]. En general, estos juegos se aplican a dos jugadores, para evitar considerar matrices multidimensionales.

Definición 3.1.1. *Un juego de matriz es un juego entre dos jugadores definido por una matriz de recompensas* $(R^1, R^2)_{i,j}$*, donde:*

- *Los dos jugadores eligen acciones* $a^1 \in \mathcal{A}^1$ *y* $a^2 \in \mathcal{A}^2$ *simultáneamente.*
- *El jugador* i *recibe una recompensa* $r^i = R^i(a^1, a^2)$.

El marco teórico de los juegos de matrices permite estudiar formalmente los dilemas sociales, considerando que los jugadores deben elegir entre dos acciones: cooperar (C) o traicionar (D), recibiendo una recompensa que depende de la acción propia y de la del oponente. Cada combinación de acciones genera una de las siguientes cuatro situaciones:

- R: Recompensa mutua por cooperar (C, C).
- S: Explotación por cooperar con un traidor (C, D).
- T: Tentación de traicionar contra un cooperador (D, C).
- P: Castigo mutuo por traicionar (D, D).

De esta manera, puede definirse un juego de matriz de dos jugadores que sea un dilema social [37], es decir, que presente las características de que los jugadores pueden beneficiarse individualmente de actuar de forma egoísta, pero si los dos lo hicieran, el resultado sería peor para ellos que si cooperasen. Estos juegos han sido usados extensamente para estudiar la emergencia de cooperación, castigo, reputación y reciprocidad en entornos controlados [2, 12].

Definición 3.1.2. *Un juego de matriz binario es un dilema social (MGSD, del inglés Matrix Game Social Dilemma) si las recompensas asociadas a las acciones C (cooperar) y D (traicionar) satisfacen las siguientes desigualdades:*

(1) $R > P$*, la cooperación mutua es preferible a la traición mutua.*

(2) $R > S$*, la cooperación mutua es preferible a ser explotado.*

(3) $2R > T + S$*, la cooperación mutua supera en valor esperado a una mezcla aleatoria entre cooperar y traicionar.*

(4) Además, se requiere que se cumpla al menos una de las siguientes:

- *Codicia:* $T > R$*, la tentación de traicionar supera al beneficio de la cooperación.*
- *Miedo:* $P > S$*, se prefiere la traición mutua a ser explotado.*

Estas desigualdades capturan la esencia del dilema: aunque la cooperación mutua sea socialmente óptima, existen incentivos individuales para traicionar. Además, la tercera condición asegura que la cooperación sea preferible incluso bajo incertidumbre sobre la acción del otro jugador. Por último, la variabilidad en la cuarta condición define los tres ejemplos clásicos de dilemas sociales: juego de la gallina (codicia), caza del ciervo (miedo) y dilema del prisionero (codicia y miedo).

3.1.1. Juego de la gallina

El juego del gallina representa un dilema en el que dos individuos se enfrentan en una situación de riesgo creciente, donde ceder (C, cooperar) evita el desastre pero implica perder estatus, mientras que persistir (D, traicionar) puede conllevar beneficios, pero también consecuencias catastróficas si ambos insisten. Este modelo fue formalizado en el contexto de la teoría de juegos por Schelling [53].

> *Dos automóviles se dirigen uno hacia el otro a gran velocidad en una carretera estrecha. Si uno se desvía, se le considera un "gallina" y pierde. Si ambos se desvían, ambos son "gallinas" y pierden, pero no hay accidente. Si ninguno se desvía, tienen un accidente.*

La matriz de recompensas para este juego se puede representar como sigue, donde los valores indican el beneficio recibido por cada jugador (a mayor número, mayor recompensa):

Jugador 1 \ Jugador 2	Desviarse (C)	Mantenerse (D)
Desviarse (C)	$(2, 2)$	$(0, 3)$
Mantenerse (D)	$(3, 0)$	$(-1, -1)$

Tabla 3.1: *Matriz de recompensas del juego del gallina.*

En este juego hay dos equilibrios de Nash puros asimétricos: (C, D) y (D, C). El peor resultado (D, D) implica un accidente, mientras que la cooperación mutua (C, C) es segura pero no óptima individualmente. Se trata de un dilema donde la codicia puede impedir alcanzar el óptimo colectivo, que sería cooperar. Los valores relevantes son $R = 2$, $S = 0$, $T = 3$ y $P = -1$, por lo que se cumplen las condiciones (1), (2) y (3) de la definición 3.1.2, y en la condición (4) se cumple la codicia: $T > R$, pero no se cumple el miedo: $P \leq S$.

3.1.2. Caza del ciervo

Inspirado por una parábola atribuida a Rousseau [49], el dilema de la caza del ciervo describe una situación donde la cooperación ofrece la mejor recompensa, pero solo si ambos jugadores confían plenamente en la colaboración del otro. El escenario es el siguiente:

> *Dos cazadores se adentran en el bosque preparados para cazar, con dos opciones: perseguir juntos a un ciervo, o individualmente atrapar liebres.*

> *El ciervo proporciona más carne que una jornada de caza de liebres, pero requiere la cooperación de ambos para ser abatido. En cambio, cada cazador puede atrapar liebres por su cuenta.*

La matriz de recompensas para este juego se puede representar como sigue, donde los valores indican el beneficio recibido por cada jugador (a mayor número, mayor recompensa):

Jugador 1 \ Jugador 2	Ciervo (C)	Liebre (D)
Ciervo (C)	$(2, 2)$	$(0, 1)$
Liebre (D)	$(1, 0)$	$(1, 1)$

Tabla 3.2: *Matriz de recompensas del dilema del ciervo.*

En este juego, existen dos equilibrios de Nash puros: (C, C) y (D, D). El primero es eficiente y socialmente deseable, pero arriesgado: si un jugador decide cazar el ciervo y el otro no coopera, el primero obtiene cero. Se trata de un dilema donde el miedo a quedarse sin nada impide alcanzar el óptimo colectivo. En términos de recompensas, identificamos $R = 2$, $P = 1$, $S = 0$ y $T = 1$, por lo que se cumplen las condiciones (1), (2) y (3) de la definición 3.1.2, y en la condición (4) se cumple solo el miedo: $P > S$, pero no se cumple la codicia: $T \leq R$.

3.1.3. Dilema del prisionero

Formalizado por Tucker [62], dos prisioneros son arrestados por el mismo crimen y puestos en celdas separadas, sin posibilidad de comunicarse. La fiscalía tiene pruebas sólidas solo para condenarlos por un delito menor, lo que implica 1 año de prisión para cada uno. No obstante, se les ofrece un trato:

> *Si uno testifica (D, traiciona) y el otro permanece en silencio (C, coopera), el traidor queda libre y el otro recibe 3 años de condena. Si ambos traicionan (D, D), se reparten la culpa y reciben 2 años cada uno. Si ambos guardan silencio (C, C), reciben solo 1 año cada uno por el delito.*

Este escenario se puede representar con la siguiente matriz de recompensas, donde los valores representan los años de condena en negativo (a mayor número, mejor resultado para el jugador):

Jugador 1 \ Jugador 2	Silencio (C)	Testificar (D)
Silencio (C)	$(-1, -1)$	$(-3, 0)$
Testificar (D)	$(0, -3)$	$(-2, -2)$

Tabla 3.3: *Matriz de recompensas del dilema del prisionero, donde los valores indican años de condena (en negativo).*

En el dilema del prisionero, ambos jugadores tienen un incentivo dominante a traicionar, lo que lleva a un único equilibrio de Nash subóptimo $(-2, -2)$, pues

$(-1, -1)$ sería mejor para ambos pero no es un equilibrio de Nash (si cualquiera cambia, mejora su situación individual). Además, se puede comprobar que $R = -1$, $S = -3$, $T = 0$ y $P = -2$, luego se cumplen las condiciones (1), (2) y (3) de la definición 3.1.2, y en la condición (4) se cumplen simultáneamente la codicia ($T > R$) y el miedo ($P > S$).

3.2. Dilemas sociales secuenciales

La teoría de juegos ha sido fundamental para modelar interacciones estratégicas entre agentes racionales. Sin embargo, los juegos de matrices clásicos, como el dilema del prisionero, representan decisiones estáticas y simultáneas, lo que limita su aplicabilidad en escenarios más complejos y realistas. Para abordar esta limitación, Leibo et al. [28] han introducido el concepto de dilemas sociales secuenciales (SSD, del inglés *Sequential Social Dilemmas*), que extienden los dilemas sociales a entornos donde las decisiones y sus consecuencias se desarrollan a lo largo del tiempo, permitiendo una representación más fiel de las interacciones en entornos dinámicos y parcialmente observables.

En estos SSD, la cooperación y la traición no son acciones puntuales, sino políticas que los agentes deben aprender y adaptar en función de las observaciones y recompensas recibidas. Este enfoque permite captar la complejidad de las interacciones en entornos donde las decisiones tienen efectos a largo plazo y donde la coordinación entre agentes es crucial para maximizar el bienestar colectivo.

Definición 3.2.1. *Un dilema social secuencial es un juego de Markov multiagente (definición 2.2.1) donde existe un conflicto estructural entre:*

- *La política óptima individual π^i, que maximiza la función de valor individual $V^i\langle \pi^i, \boldsymbol{\pi}^{-i} \rangle$.*
- *Una política socialmente óptima $\boldsymbol{\pi}^{soc}$, que maximiza una métrica de bienestar social, típicamente el promedio del valor: $\sum_i V^i_{\boldsymbol{\pi}}$, o el mínimo del valor para todos los agentes:* $\text{mín}_i\, V^i_{\boldsymbol{\pi}}$.

En resumen, se dice que un juego de Markov es un SSD si existen estados s y políticas cooperativas/traidoras tales que las recompensas satisfacen las desigualdades de un dilema social clásico (véase la definición 3.1.2). Aun así, la diferencia clave es que en un SSD estas desigualdades emergen de dinámicas de largo plazo en el entorno, no de una matriz fija de una sola jugada.

A modo de ejemplo, Leibo et al. [28] han construido un SSD a partir del dilema clásico de la caza del ciervo, denominado *Wolfpack*. La dinámica del juego consiste en que dos agentes cazan una presa: un lobo solitario puede atraparla y recibe una recompensa baja, pero, si ambos colaboran, reciben una recompensa mayor por proteger mejor la presa. La analogía con el dilema de la caza del ciervo es clara: cooperar brinda gran beneficio conjunto, mientras que traicionar conduce a una menor paga.

De esta manera, los agentes deben desarrollar estrategias temporales, incluyendo memoria, aprendizaje y adaptación. Los incentivos a cooperar o traicionar no siempre son evidentes a corto plazo, y pueden requerir aprendizaje por exploración, por lo que los agentes enfrentan un dilema más complejo: cooperar puede implicar ceder recompensas a corto plazo a cambio de un entorno más beneficioso a largo plazo. Este tipo de dilemas plantea desafíos importantes para el aprendizaje por refuerzo, ya que los mecanismos clásicos de exploración-explotación no siempre conducen a resultados cooperativos estables. Además, surgen fenómenos sociales emergentes como la reciprocidad condicional, el castigo, el establecimiento de normas o incluso el colapso de recursos compartidos [20].

3.2.1. Cooperación en dilemas sociales secuenciales

Dada la estructura propia de los SSD, la emergencia de la cooperación no está asegurada y, en caso de aparecer, su estabilidad a lo largo del tiempo está en duda. De manera natural, los agentes de aprendizaje por refuerzo no tienden a cooperar a no ser que sea muy favorable, pues requiere una mayor complejidad y coordinación que actuar individualmente. Para solventar este problema, el método más generalizado es el modelado de recompensas (*reward shaping* en inglés), que consiste en dividir la recompensa obtenida por cada agente en una componente extrínseca —parte aportada por el entorno— y otra componente intrínseca —la motivación interna del agente— [44]. De esta forma, la recompensa inmediata del agente i viene dada por:

$$r^i := r^i_{ex} + r^i_{in}. \tag{3.1}$$

Utilizando esta caracterización de la recompensa puede fomentarse la cooperación si se utiliza adecuadamente la recompensa intrínseca. Así, Chen y Kempe [8] utilizan el bienestar general (SW, del inglés *social welfare*) para incentivar a los agentes a actuar en pos del bien común. Este método, denominado "altruismo", se caracteriza por un parámetro real $\beta \in [0, 1]$ que determina la importancia de cada componente de la recompensa, quedando:

$$r^i = (1 - \beta) \cdot r^i_{ex} + \frac{\beta}{N} \cdot SW(\boldsymbol{r}_{ex}), \tag{3.2}$$

donde se ha utilizado el bienestar general, SW, definido por:

$$SW(\boldsymbol{r}_{ex}) := \frac{1}{N} \sum_{i=1}^{N} r^i_{ex}. \tag{3.3}$$

Siguiendo la idea de caracterizar la recompensa en distintas partes, Hughes et al. [20] separan la parte intrínseca del altruismo de la ecuación (3.2) anterior en dos partes: un primer término de culpa cuando un agente supera la media, y uno segundo de envidia cuando un agente tiene un rendimiento inferior a la media. De esta forma, cualquier forma de desigualdad resulta negativa para la recompensa, recibiendo este método el nombre de "aversión a la inequidad". La recompensa intrínseca toma la forma:

$$r^i_{in} := \frac{-\alpha}{N-1} \sum_{j \neq i} \text{máx}(r^j_{ex} - r^i_{ex}, 0) - \frac{\beta}{N-1} \sum_{j \neq i} \text{máx}(r^i_{ex} - r^j_{ex}, 0), \tag{3.4}$$

donde α y β son hiperparámetros de control sobre la importancia de la culpa y la envidia, respectivamente. Esta idea también ha sido utilizada en otros trabajos, como McKee et al. [39], aunque en este caso con el nombre de "reputación".

Otro enfoque destacable es el propuesto por Jaques et al. [24], basado en la noción de influencia social. La idea principal consiste en incentivar aquellas acciones que afectan significativamente a las decisiones de los demás agentes, introduciendo para ello una recompensa intrínseca definida mediante la divergencia de Kullback-Leibler (KL) entre la política del agente j condicionada a la acción del agente i, es decir, $\pi^j(s, a^j|a^i)$, y la política marginal del agente j, $\pi^j(s, a^j)$:

$$r_{in}^i := \sum_{j \neq i} D_{KL}[\pi^j(s, a^j|a^i), \pi^j(s, a^j)], \tag{3.5}$$

donde la divergencia de Kullback-Leibler entre dos distribuciones de probabilidad P y Q definidas sobre una variable aleatoria discreta se expresa como [26]:

$$D_{KL}[P, Q] = \sum_i P(i) \cdot \ln\left(\frac{P(i)}{Q(i)}\right). \tag{3.6}$$

Finalmente, generalizando las posibles actitudes de un agente en función de sus recompensas, McKee et al. [38] introducen el concepto de orientación del valor social (SVO, del inglés *Social Value Orientation*). Este marco asume que cualquier actitud puede modelarse como una combinación lineal entre la recompensa extrínseca del agente, r_{ex}^i, y la recompensa media del resto de agentes, $r_{ex}^{-i} := \frac{1}{N-1} \sum_{j \neq i} r_{ex}^j$. Representando ambos valores como los ejes del plano $\mathbb{R}^2$, la actitud del agente se visualiza como un punto sobre una circunferencia centrada en el origen (véase la figura 3.1).

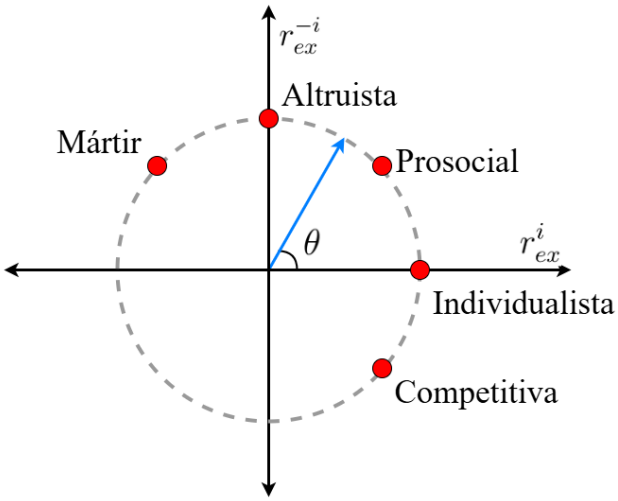


Figura 3.1: *Representación geométrica de la orientación del valor social (SVO). El eje horizontal representa la recompensa extrínseca del agente r_{ex}^i, mientras que el eje vertical representa la recompensa media del resto de agentes r_{ex}^{-i}. Cada actitud posible del agente se representa como un punto sobre la circunferencia, y se caracteriza por el ángulo de recompensa θ, que determina la proporción relativa entre recompensas propias y ajenas.*

Desde un punto de vista matemático, la distribución relativa de recompensas define la actitud del agente, caracterizada por el ángulo de recompensa (*reward*

angle en inglés):

$$\theta(\boldsymbol{r}_{ex}) = \arctan\left(\frac{r_{ex}^{-i}}{r_{ex}^{i}}\right). \tag{3.7}$$

Este ángulo es invariante respecto a la magnitud del vector $\boldsymbol{r}$ y determina la proporción relativa entre beneficio propio y ajeno. Así, se define el SVO del agente como su ángulo preferido $\theta^i SVO$, es decir, aquel que describe la distribución de recompensas a la que aspira. Esta formulación permite una caracterización matemática de diferentes actitudes: altruismo ($\theta \sim 90°$), individualismo ($\theta \sim 0°$) o competitividad ($\theta \sim -45°$), entre otras (véase de nuevo la figura 3.1).

Una consecuencia importante de esta formulación es que la recompensa total del agente puede interpretarse como su recompensa extrínseca penalizada por la discrepancia angular entre su SVO y la distribución real, ponderada mediante un parámetro β que regula la sensibilidad social del agente:

$$r^i := r_{ex}^i - \beta \cdot \left|\theta_{SVO}^i - \theta(\boldsymbol{r}_{ex})\right|. \tag{3.8}$$

Simplificando esta idea tanto matemática como conceptualmente, pero manteniendo su representación visual, se propone que la actitud de un agente depende de cómo se distribuye su recompensa en función de r_{ex}^i (recompensa propia) y r_{ex}^{-i} (recompensa ajena), ponderadas mediante dos parámetros reales α y β que satisfacen $\alpha^2 + \beta^2 = 1$:

$$r^i = \alpha \cdot r_{ex}^i + \beta \cdot r_{ex}^{-i}. \tag{3.9}$$

Bajo esta formulación, la circunferencia representada en la figura 3.1 sigue siendo válida, aunque ahora tiene radio unitario. Las diferentes actitudes del agente quedan determinadas por los valores de α y β. En particular, se proponen cuatro tipos fundamentales de actitud (véase la figura 3.2): cooperativa ($\alpha, \beta > 0$), competitiva ($\alpha > 0$ y $\beta < 0$), sacrificada ($\alpha < 0$ y $\beta > 0$) y destructiva ($\alpha, \beta < 0$).

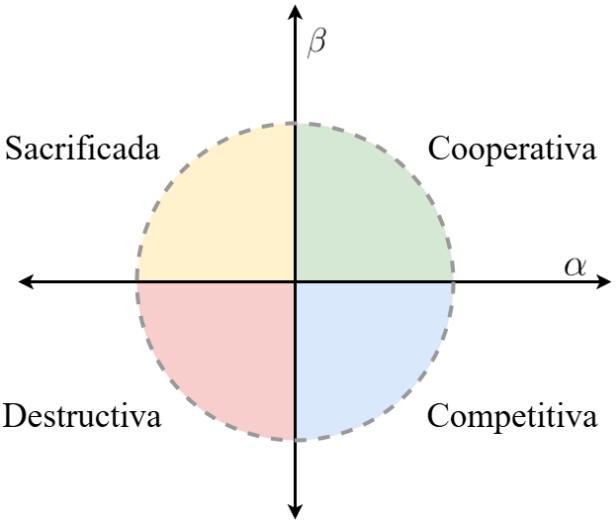


Figura 3.2: *Representación geométrica de las actitudes de un agente en función de los parámetros α y β, que ponderan las recompensas propias r_{ex}^i y ajenas r_{ex}^{-i}, respectivamente. Cada punto sobre la circunferencia unitaria describe una actitud determinada según el signo de α y β.*

CAPÍTULO 4

Estado del arte

Los sistemas de aprendizaje por refuerzo multiagente (MARL, por sus siglas en inglés) pueden exhibir comportamientos sociales como la cooperación, la competición, la equidad, el altruismo y la reciprocidad. En los últimos años se ha estudiado cómo surgen (o no) las dinámicas cooperativas en sistemas multiagente, utilizando diversos paradigmas de aprendizaje y diseños de entornos.

En términos generales, los métodos se dividen en dos tipos de entrenamiento (véase la sección 2.2.3): CTDE, en el que los agentes pueden aprovechar la información global o las críticas compartidas durante el aprendizaje, y DTDE, en el que cada agente aprende de forma independiente con sólo observaciones locales. Los enfoques CTDE (por ejemplo, métodos actor-crítico con crítico centralizado) suelen producir un comportamiento de equipo coherente en tareas totalmente cooperativas, mientras que los enfoques DTDE (aprendices independientes) dependen a menudo de incentivos sociales intrínsecos para coordinarse eficazmente, como se puede observar en la tabla 4.1.

Los dilemas sociales (dilema del prisionero, caza del ciervo o juego de la gallina) y los *grid-world games*, i.e. juegos de mundo cerrado organizado y cuadriculado (búsqueda de comida, tareas de cocina o limpieza conjunta), son *benchmarks*[1] habituales para estudiar la cooperación emergente, como se puede comprobar en la tabla 4.2.

A continuación, se repasan los principales avances teóricos y empíricos de la última década, centrando el foco en cómo diversos algoritmos MARL, estructuras de recompensa y diseños de entornos influyen en la aparición de comportamientos cooperativos frente a egoístas, y resumiendo la configuración, el paradigma de aprendizaje y los fenómenos sociales observados de cada trabajo citado.

[1]En el contexto de la teoría de juegos, *benchmark* (o prueba de rendimiento) se refiere a una técnica o herramienta que se utiliza para evaluar el rendimiento de un sistema o componente.

4.1. Dinámicas de entrenamiento: *DTDE vs CTDE*

Las primeras investigaciones sobre MARL distinguían entre aprendices independientes y métodos de aprendizaje conjunto [61]. Posteriormente, la clasificación de los tipos de aprendizaje ha tendido a distinguir entre DTDE, formalizando el concepto de aprendices independientes, y CTDE, describiendo el concepto de aprendizaje conjunto [16].

En el marco de DTDE, Leibo et al. [28] utilizan DQN para entrenar a cada agente con su propia red neuronal, aplicándolo a *sequential social dilemmas* (dilemas sociales secuenciales), un tipo de dilema en el que las decisiones individuales afectan no solo al estado actual del entorno, sino también a las decisiones futuras de otros agentes (véase la sección 3.2). Sin embargo, los enfoques DTDE presentan limitaciones importantes en entornos cooperativos, debido a problemas como la no estacionariedad, la asignación de créditos y la presencia de incentivos egoístas [45]. Para paliar estos problemas, se han propuesto mecanismos adicionales como la modificación de las recompensas, la motivación intrínseca o formas de comunicación emergente.

Por el contrario, los enfoques CTDE han demostrado ser más eficaces en tareas cooperativas complejas. Por ejemplo, Samvelyan et al. [52] aplicaron los algoritmos QMIX y COMA a la microgestión de unidades en SMAC, una versión multiagente del videojuego *StarCraft II*. QMIX ha mostrado un rendimiento notable en tareas que requieren coordinación precisa, como el combate grupal en tiempo real [48], mientras que MADDPG ha sido eficaz en simulaciones físicas donde varios robots colaboran para alcanzar objetivos comunes [34]. En general, los algoritmos de la familia CTDE (como VDN, QMIX, COMA y MADDPG) logran mayores recompensas en entornos totalmente cooperativos, superando con frecuencia a los métodos DTDE [71].

Recientemente, numerosos trabajos han tratado de cerrar la brecha entre ambos paradigmas mediante incentivos sociales o estructuras de comunicación más sofisticadas. Por ejemplo, Jaques et al. [24] introducen una *recompensa por influencia* que incentiva a un agente a tomar acciones que afectan significativamente a las decisiones de otros (véase la ecuación (3.5)). Esta recompensa intrínseca descentralizada mejora la coordinación y favorece el surgimiento de protocolos de comunicación emergente, incluso sin un crítico centralizado.

Siguiendo esta línea, Feng et al. [14] proponen HC-MARL (*Hierarchical Consensus Multi-Agent Reinforcement Learning*), un modelo que introduce mecanismos de consenso jerárquico. En este enfoque, los agentes aprenden representaciones compartidas del entorno a partir de observaciones locales, mediante aprendizaje contrastivo y sin necesidad de comunicación explícita. El modelo incorpora dos niveles de consenso: uno a corto plazo, que facilita la coordinación inmediata, y otro a largo plazo, que permite una planificación coherente a lo largo del tiempo. Esta arquitectura ha mostrado mejoras notables en la coordinación de sistemas multi-robot, especialmente en entornos donde la comunicación directa es limitada o costosa.

Además, Kong et al. [25] proponen LASE (del inglés *Learning to balance Altruism and Self-interest based on Empathy*), un algoritmo de aprendizaje por refuerzo multiagente descentralizado diseñado para abordar dilemas sociales con motivación mixta,

donde los agentes deben equilibrar el altruismo y el interés propio. Inspirado en la empatía cognitiva, LASE introduce un modelo computacional que infiere relaciones sociales entre agentes mediante razonamiento contrafactual y un módulo de toma de perspectiva. Este enfoque permite a los agentes asignar recompensas a otros de manera adaptativa, basándose en la contribución de cada co-agente al bienestar colectivo, fomentando así la cooperación sin comprometer la equidad ni exponerse a la explotación.

Técnicamente, LASE se implementa en un marco descentralizado donde cada agente aprende una política independiente utilizando el método actor-crítico. La arquitectura de la red neuronal de cada agente incluye dos capas convolucionales para procesar las observaciones, una capa LSTM (*long short-term memory*) para capturar información temporal y varias capas completamente conectadas. El módulo de toma de perspectiva simula la observación de otros agentes a partir de la observación local del agente, permitiendo estimar las políticas de los co-agentes. El módulo de inferencia de relaciones sociales evalúa la "amistad" o impacto de cada co-agente en el retorno, comparando el valor estimado (*Q-value*) de la acción conjunta actual con un valor base contrafactual que marginaliza la acción del co-agente. Basándose en estas relaciones, el módulo de donaciones determina la cantidad de recompensa que el agente debe compartir con otros.

Trabajo	**Tipo**	**Algoritmo**	**Contribuciones**	**Limitaciones**
Leibo et al. [28]	DTDE	DQN	Modelado de dilemas secuenciales, con consecuencias temporales	Egoísmo, *credit assignment problem*
Foerster et al. [16]	CTDE	COMA	Uso de críticos centralizados para asignar créditos (contrafactuales)	Requiere acceso centralizado a información
Rashid et al. [48]	CTDE	QMIX	Factorización de valor en aprendizaje cooperativo	Puede ser limitado en entornos parcialmente observables
Lowe et al. [34]	CTDE	MADDPG	Observadores centralizados y políticas descentralizadas para escenarios mixtos y cooperativos	Costoso computacionalmente
Jaques et al. [24]	CTDE	PPO + recompensa social	Introducción de la recompensa por influencia social, que mejora la coordinación sin comunicación explícita	Difícil de sintonizar; comportamiento no siempre estable
Feng et al. [14]	CTDE	HC-MARL	Coordinación sin comunicación; aprendizaje de consenso con observaciones locales	Complejidad arquitectónica

Tabla 4.1: *Resumen de algoritmos clave en aprendizaje multiagente.*

4.2. Dilemas sociales y emergencia de la cooperación

Una clase fundamental de problemas para estudiar la cooperación son los dilemas sociales, en los que la racionalidad individual entra en conflicto con el bienestar colectivo. Entre los ejemplos tradicionales de la teoría de juegos se encuentran el dilema del prisionero, la gallina y la caza del ciervo (véase la sección 3.1). Sin embargo, la investigación en MARL se centra en las extensiones secuenciales de estos dilemas, pues los entornos multiagente se desarrollan a lo largo del tiempo y tanto las recompensas como las decisiones son dinámicas. Leibo et al. [28] introducen los dilemas sociales secuenciales (SSD, véase la sección 3.2) ampliando los juegos de matriz a juegos de Markov dinámicos: proponen el juego *Gathering*, en el que los agentes compiten por recursos renovables, y el juego *Wolfpack*, que requiere que los agentes capturen presas conjuntamente.

Este tipo de juegos se denominan *grid-world*, y son entornos discretos representados como una cuadrícula en la que los agentes se mueven y toman decisiones secuenciales. En ellos, cada agente utiliza un DQN (véase la sección 2.1.1) independiente —totalmente descentralizado, DTDE— y los autores observan que la abundancia de recursos y los parámetros del entorno afectan críticamente al comportamiento emergente. En condiciones de gran abundancia, los agentes tienden a recolectar de forma egoísta, mientras que la escasez de recursos los incentiva a cooperar (por ejemplo, compartir recursos o coordinar cacerías), ilustrando así cómo la estructura secuencial puede inducir o dificultar la cooperación. Este trabajo pone de relieve que la dinámica temporal puede cambiar drásticamente la naturaleza de la cooperación.

Sobre la base de estas ideas, Hughes et al. [20] examinan la aversión a la inequidad en los SSD. Incorporan las preferencias sociales en la recompensa de cada agente penalizando las desigualdades (siguiendo los modelos de Fehr-Schmidt [13], véase la ecuación (3.4)) y entrenan agentes por refuerzo independientes en el juego *Cleanup*. En este entorno, los agentes recolectan manzanas para obtener recompensas, pero deben limpiar un río contaminado para que las manzanas vuelvan a crecer, lo que introduce un conflicto entre el beneficio individual y el bien común. Los resultados de este trabajo muestran que los agentes con recompensas reacias a la desigualdad aprenden a cooperar, para lo cual castigan a los traidores haciendo que paguen para aumentar el beneficio del grupo. Así, la adición de preferencias sociales intrínsecas permite a los agentes escapar del equilibrio de Nash y lograr resultados mutuamente beneficiosos.

Por otro lado, y como se ha analizado previamente, Kong et al. [25] abordan directamente los desafíos que presentan los dilemas sociales de motivación mixta en entornos multiagente descentralizados. El algoritmo que proponen, LASE, da solución a esta tensión, permitiendo a los agentes aprender a equilibrar el altruismo y el interés propio a través de mecanismos de empatía computacional. Los resultados experimentales muestran que los agentes entrenados con LASE cooperan de forma estable incluso en contextos de riesgo mutuo, como en la caza del ciervo secuencial, y logran repartir equitativamente las cargas y beneficios colectivos, como en *Cleanup*. A diferencia de enfoques puramente egoístas o altruistas, LASE genera un equilibrio dinámico y contextualizado entre el interés propio y la prosocialidad, promoviendo soluciones robustas frente a la explotación y mejorando la equidad a nivel grupal.

En conjunto, el trabajo demuestra que dotar a los agentes de una forma de empatía operativa puede transformar radicalmente la dinámica de cooperación en dilemas sociales complejos.

La reputación y la reciprocidad son también mecanismos de cooperación. Smit and Santos [58] estudian un juego de donación con agentes de *Q-learning* independientes que mantienen normas sociales y reputaciones. Utilizando un marco de reciprocidad indirecta, cada agente asigna puntuaciones de reputación a los demás en función de las acciones observadas, de acuerdo con diferentes normas sociales. Así, se descubre que el juicio severo (*stern-judging* en inglés) —norma social que aprueba a un agente si ayuda a alguien de buena reputación o rechaza a alguien de mala reputación, pero desaprueba si ayuda a alguien con mala reputación o rechaza a alguien de buena reputación— conduce tanto a la cooperación como a la equidad en la población, mientras que otras normas pueden producir injusticia o traición. En esencia, la introducción de un sistema de reputación permite a los agentes descentralizados corresponder a las buenas acciones y mantener la cooperación.

En la misma línea, McKee et al. [39] utilizan el juego *Cleanup* para investigar cómo la motivación intrínseca por la reputación puede fomentar la cooperación en dilemas sociales secuenciales. Los autores desarrollan un modelo computacional basado en el método actor-crítico, en el que cada agente recibe una recompensa total compuesta por una parte extrínseca —por la recolección de manzanas— y una parte intrínseca relacionada con la reputación (véase la ecuación (3.4)). Esta recompensa intrínseca penaliza las discrepancias entre la contribución individual al bien común, i.e. la limpieza del río, y la contribución promedio del grupo, incentivando así comportamientos cooperativos. Los resultados muestran que, cuando los agentes pueden identificar a los demás y rastrear sus reputaciones, emergen estrategias de coordinación temporal, como la rotación no territorial en las tareas de limpieza, permitiendo una distribución equitativa de las cargas y beneficios, manteniendo la sostenibilidad del entorno. En cambio, en condiciones de anonimato, donde no es posible rastrear las contribuciones individuales, la cooperación se deteriora significativamente.

Por último, Radke et al. [47] introducen la idea de un "credo", i.e. un equipo en el que los agentes tienen objetivos parcialmente alineados. Así, estudian distintos dilemas sociales dejando que los agentes optimicen una recompensa multiobjetivo: en parte el interés del equipo y en parte el interés local. Incluso sin una alineación total de intereses, este enfoque logra una gran igualdad y una recompensa total mayor que los agentes puramente egoístas o totalmente altruistas. Concretamente, demuestran que en algunos casos no es necesario que todos los agentes sacrifiquen plenamente el interés propio para alcanzar un buen resultado grupal, sino que una ponderación adecuada puede equilibrar el bienestar grupal y la ganancia individual.

En resumen, en todos estos trabajos la aparición de la cooperación depende sensiblemente de la configuración, y el equilibrio cooperativo es difícil de conseguir. Las tareas totalmente cooperativas fomentan la maximización de la recompensa del equipo [16], mientras que los dilemas sociales con tentaciones de traición suelen requerir incentivos adicionales para la cooperación [28]. Además, dichos incentivos sociales —como la aversión a la desigualdad [20] o la reputación [58]— pueden permitir la cooperación entre agentes descentralizados. Por último, la mezcla de agentes que cooperan y agentes que compiten puede llevar a la formación espontánea de subgrupos o roles [24, 47].

Efecto de las distintas actitudes en los dilemas sociales

Una línea de investigación emergente en el estudio de los dilemas sociales secuenciales se centra en la incorporación de la *Social Value Orientation* (SVO), un concepto de la psicología social que cuantifica el grado en que un individuo prioriza el bienestar propio frente al de los demás —véanse la figura 3.1 y la ecuación (3.8)—. Este enfoque ha sido explorado en profundidad por McKee et al. [38] y ha influido en trabajos posteriores que adaptan esta idea a contextos diversos.

McKee et al. [38] implementan agentes de aprendizaje por refuerzo con diferentes orientaciones sociales (SVOs), representando una gama de comportamientos desde egoístas hasta altruistas. En entornos de dilemas sociales secuenciales, como variantes del dilema del prisionero, descubren que la diversidad en SVO conduce a una variedad significativa de comportamientos cooperativos. Además, los agentes entrenados en poblaciones heterogéneas desarrollan políticas más generalizadas y efectivas en comparación con aquellos en poblaciones homogéneas, sugiriendo que la diversidad social puede mejorar la adaptabilidad y la cooperación en entornos complejos.

Trabajo	**Mecanismo cooperativo**	**Tipo de agente**	**Dinámica social**	**Limitaciones**
Leibo et al. [28]	Escasez de recursos	DQN descentralizados	Cooperación emerge de la dinámica del entorno	No hay control explícito sobre motivaciones
Hughes et al. [20]	Aversión a la desigualdad	Recompensas incluyendo preferencias sociales	Cooperación mediante penalización de la inequidad	Requiere sintonía específica de parámetros
Kong et al. [25]	Empatía computacional	Algoritmo propio (LASE)	Equilibrio dinámico entre altruismo y egoísmo	Difícil de interpretar y analizar
Smit and Santos [58]	Normas sociales y reputación	*Q-learning*	Reciprocidad indirecta basada en juicios sociales	Depende del diseño de normas complejas
McKee et al. [39]	Reputación como motivación	Actor-crítico	Coordinación rotativa para repartir tareas	Colapso de cooperación sin trazabilidad social
Radke et al. [47]	Recompensas parcialmente alineadas	Enfoque multiobjetivo con pesos	Equilibrio equipo-individuo	Falta de control explícito sobre actitudes
McKee et al. [38]	Diversidad en la orientación de valor social (SVO)	Agentes con distintas SVOs	Mayor generalización en poblaciones heterogéneas	Sin análisis del efecto mutuo entre actitudes

Tabla 4.2: *Resumen comparativo de trabajos sobre cooperación en entornos multiagente.*

Extendiendo este trabajo, Xiang et al. [70] exploran métodos de aprendizaje autosupervisado en entornos multiagente, comparando enfoques centralizados y descentralizados. Aunque no se centran exclusivamente en SVO, identifican la importancia de las motivaciones intrínsecas para fomentar estrategias sociales adaptativas en agentes autónomos que operan con información limitada.

En otra dirección distinta, Zhang et al. [77] proponen un sistema de evaluación de alineación de valores para *Large Language Models* (LLMs), utilizando SVO como una métrica para medir la racionalidad de valor. Asignan diferentes SVOs a los modelos y evalúan si sus respuestas se alinean con las orientaciones sociales inducidas: una herramienta para analizar cómo los modelos reflejan diversas preferencias sociales.

Estos estudios destacan la versatilidad del concepto de SVO para modelar y evaluar comportamientos cooperativos en agentes artificiales, tanto en entornos de aprendizaje por refuerzo como en modelos de lenguaje, subrayando su relevancia en el diseño de sistemas que interactúan en contextos sociales complejos.

4.3. Aplicaciones reales de la cooperación

Más allá de los entornos simulados anteriormente descritos, la investigación en aprendizaje por refuerzo multiagente (MARL, por sus siglas en inglés) ha comenzado a centrarse en sistemas cooperativos del mundo real, con aplicaciones destacadas en robótica, transporte y otros dominios. Estas aplicaciones presentan complejidades ausentes en los entornos simulados —como ruido, asincronía o restricciones de seguridad— que requieren enfoques metodológicos específicos.

Uno de los ámbitos más explorados es la robótica cooperativa. Labiosa and Hanna [27] entrenan políticas en entornos simulados y las transfieren posteriormente a equipos de robots físicos que juegan al fútbol de forma cooperativa. En su enfoque, se omiten numerosos detalles de bajo nivel, manteniéndose únicamente los aspectos tácticos esenciales. De forma notable, las políticas MARL obtenidas alcanzan un rendimiento comparable al de una estrategia *RoboCup*[2] diseñada manualmente para robots reales (véase la figura 4.1). Este resultado pone de manifiesto que MARL puede inducir coordinación emergente —como pases o posicionamiento— incluso cuando el entrenamiento se realiza sobre modelos altamente simplificados.

En una línea similar, Yu et al. [72] abordan la ejecución asíncrona en entornos de exploración cooperativa, dada la imposibilidad, en contextos reales, de que los robots actúen de forma perfectamente sincronizada. Para ello, desarrollan el explorador de coordinación asíncrona (ACE, del inglés *Asynchronous Coordination Explorer*), un mecanismo que introduce retrasos aleatorios en la ejecución de acciones durante el entrenamiento, con el fin de robustecer las políticas frente a la asincronía. Los experimentos, tanto en simuladores tipo *grid-world* como en plataformas físicas, muestran que ACE reduce significativamente el tiempo de exploración en comparación con enfoques MARL síncronos estándar.

[2]Una estrategia *RoboCup* hace referencia al conjunto de comportamientos coordinados que implementan equipos de robots autónomos en la competencia internacional del mismo nombre. Estas estrategias incluyen desde la planificación de movimientos individuales hasta la coordinación táctica entre múltiples robots con el objetivo de maximizar el rendimiento colectivo.

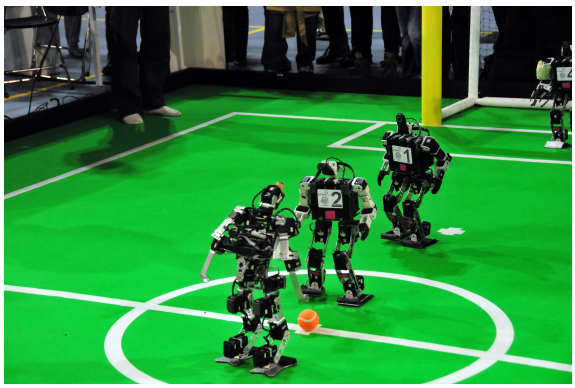


Figura 4.1: *Robots cooperando en un partido de fútbol autónomo durante la RoboCup 2013 en Eindhoven. Imagen de de Kok 2013, bajo licencia CC BY-SA 3.0.*

Por otro lado, la cuestión de la equidad ha sido recientemente explorada por Aloor et al. [1], motivados por aplicaciones de movilidad autónoma como el transporte compartido o la planificación de rutas en redes urbanas. A través de tareas de navegación simuladas con agentes descentralizados, los autores observan que, sin mecanismos correctivos, ciertos agentes recorren sistemáticamente mayores distancias que otros. Para mitigar esta desigualdad, se introduce un componente de equidad en la función de recompensa, lo que permite a los agentes aprender asignaciones de tareas más equilibradas. Como resultado, se alcanza una cobertura espacial casi perfecta y se mejoran las métricas de equidad —medidas mediante el coeficiente de variación de las distancias recorridas— con un impacto marginal en la eficiencia.

Este trabajo constituye una de las escasas contribuciones que aborda explícitamente el equilibrio entre eficiencia y equidad en entornos multiagente vehiculares, y demuestra que, mediante un diseño adecuado, MARL puede incorporar consideraciones sociales relevantes, como la distribución equitativa de la carga de trabajo. En esta línea, otros estudios han aplicado MARL a tareas de conducción cooperativa —por ejemplo, incorporación en múltiples carriles o evasión de colisiones— integrando componentes sociales en la función de recompensa. Sin embargo, este enfoque sigue en una fase inicial y, en su mayoría, continúa restringido a entornos simulados.

La cooperación también ha sido estudiada en aplicaciones como el control distribuido de enjambres de vehículos aéreos no tripulados (UAVs), la gestión de redes energéticas inteligentes y redes de sensores distribuidos. En estos casos, surgen tensiones entre enfoques de entrenamiento centralizado con ejecución descentralizada (CTDE) y enfoques completamente descentralizados (DTDE). Por ejemplo, Viseras et al. [64] proponen un marco de aprendizaje por refuerzo cooperativo que permite a un equipo de drones autónomos monitorizar frentes de incendio.

Por el contrario, Zhang et al. [74] presentan una solución CTDE para la gestión energética en comunidades inteligentes. Introducen el enfoque LSD-MADDPG (del inglés *Local Strategy-Driven Multi-Agent Deep Deterministic Policy Gradient*), que coordina el consumo energético entre edificios inteligentes, logrando una reducción significativa de los costes energéticos y una mayor estabilidad en la curva de demanda. Estos estudios convergen en una conclusión común: la cooperación tiende a emerger cuando los agentes optimizan objetivos colectivos, mientras que los agentes puramente egoístas rara vez alcanzan una coordinación eficaz.

CAPÍTULO 5

Metodología

Con la intención de contrastar las distintas dinámicas de comportamiento y aprendizaje que presentan los agentes entrenados en un sistema multiagente, se ha implementado un entorno simulado en el que dos agentes deberán recoger monedas en un espacio compartido. El objetivo de este estudio ha sido analizar cómo distintos tipos de recompensas y actitudes sociales influyen en el comportamiento emergente de dichos agentes, enfrentando también posibles dilemas sociales. En particular, se ha estudiado cómo la cooperación y el conflicto cambian bajo dos situaciones distintas: un entorno que simula el dilema del prisionero y otro entorno sin dilema.

Para ello, se ha utilizado el entorno *CoopCoins* [23], una modificación del entorno *Coin Game* de la librería `JaxMARL` [51], que ha sido ampliamente utilizado en estudios previos para modelar escenarios cooperativos o competitivos entre agentes [29, 15, 35]. Esta adaptación modifica la dinámica y el sistema de recompensas para reflejar diferentes actitudes y estructuras sociales. Además, se han introducido distintas métricas que miden tanto el rendimiento individual de cada agente como el colectivo.

5.1. Entorno simulado: *CoopCoins*

El entorno *CoopCoins* [23] es una adaptación del entorno original *Coin Game*: un mapa dividido en casillas (*grid-world* en inglés) que permite simular dilemas sociales como el dilema del prisionero (véase la tabla 3.3). Esta adaptación presenta varias ventajas analíticas, pero la principal es que puede estudiarse tanto desde un punto de vista cooperativo como desde uno competitivo, englobándose en los juegos de motivación mixta (véase la figura 2.5). La disposición visual del entorno puede observarse en la figura 5.1.

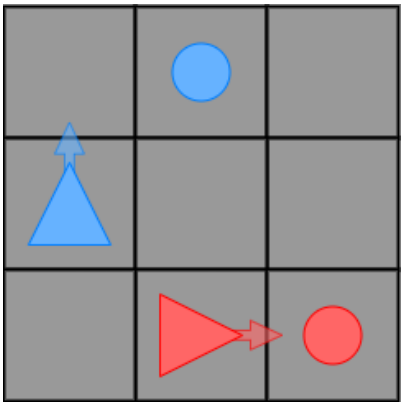


Figura 5.1: *Visualización del entorno CoopCoins, donde los agentes (triángulos) recogen monedas (círculos) del mismo o distinto color en un entorno tipo grid-world.*

Propuesto por primera vez por Lerer and Peysakhovich [29], *Coin Game* consiste en dos jugadores, uno rojo y otro azul, que deben recoger monedas en un mapa cuadrado —en su versión original, de tamaño 3×3—. Las monedas también están etiquetadas como rojas o azules, de forma que si un jugador coge una moneda, recibe un punto, pero si su oponente coge una moneda de su color, pierde dos puntos.

Sobre dicho entorno, Lerer and Peysakhovich [29] han probado agentes amTFT (del inglés *approximate Tit-for-Tat*) —diseñados para fomentar reciprocidad— frente a agentes egoístas, y han observado que el entrenamiento cooperativo favorece la coordinación estable. Esto es, han encontrado que en el modo egoísta los agentes terminan recogiendo monedas de todos los colores, mientras que en el modo cooperativo convergen a sólo recoger monedas de su color.

Sobre este mismo entorno, Foerster et al. [15] han probado un aprendizaje con consciencia del oponente (LOLA), aplicando métodos actor-crítico centralizados. En este estudio muestran que agentes estándar ("aprendices ingenuos" con *policy gradient*) recogen monedas indiscriminadamente, mientras que agentes LOLA aprenden a cooperar: el 80 % de las monedas que recogen son de su propio color. Es decir, aprenden a evitar las "monedas ajenas" y obtienen así mayor recompensa conjunta.

Por su parte, Lu et al. [35] han propuesto M-FOS (del inglés *Model-Free Opponent Shaping*) —método de entrenamiento que "moldea" el aprendizaje del oponente— y lo han evaluado también en el entorno *Coin Game*. En sus simulaciones, los agentes PPO entrenados juntos recogen monedas de ambos colores indiscriminadamente, mientras que los agentes M-FOS aprenden a elegir las monedas propias. Además, cuando un agente PPO entrena con otro M-FOS, aprende gradualmente a preferir sus propias monedas, reflejando un cambio en su comportamiento.

En definitiva, el entorno *Coin Game* ha sido ampliamente estudiado en la modelización de dinámicas de aprendizaje. En este trabajo se propone entonces, en lugar de comparar distintos métodos de entrenamiento, estudiar cómo la distribución de las recompensas genera un cambio en los comportamientos de los agentes, llevando a la necesidad de adaptar el entorno en uno nuevo: *CoopCoins*. Aun así, el hilo conductor es similar: inducir cooperación o explotar estrategias sociales, buscando mapear el *continuum* entre egoísmo y altruismo y observando cómo esto modifica el comportamiento bajo la presencia (o no) de un dilema social implícito.

5.1.1. Características del entorno *CoopCoins*

Este nuevo entorno, *CoopCoins* [23], consiste en un mapa de dimensión $N \times N$ en el que cada agente puede ejecutar una de cinco acciones discretas: moverse hacia arriba, abajo, izquierda, derecha o permanecer en su posición. Además, la recogida de monedas no se modela como una acción explícita, sino que ocurre automáticamente cuando un agente se desplaza a la celda ocupada por una moneda.

Si un agente se posiciona sobre una moneda, la recoge. Derivado de esa acción el agente recibe una recompensa, que depende de la matriz de pagos (véase la sección 5.1.2 siguiente), indicando las recompensas por recoger una moneda propia y por recoger una ajena. Además, si recoge una moneda ajena, el otro agente recibe una penalización, i.e. una recompensa negativa, también determinada por la matriz de pagos. De esta forma, el juego puede ser tanto cooperativo —si se establece una penalización nula— como competitivo.

El entorno ha sido adaptado a la biblioteca JAX para aprendizaje por refuerzo multiagente, lo que permite su integración con la librería `JaxMARL` [51]. Además, se ha desarrollado una versión compatible con `RLlib`, la biblioteca de Ray para aprendizaje por refuerzo [30], lo que posibilita comparar implementaciones entre marcos distintos. Ambas implementaciones están disponibles públicamente en Python.

El desarrollo del juego es episódico, cada uno con múltiples pasos internos. Así, los agentes van acumulando recompensas a medida que recogen monedas, que van reapareciendo en distintas posiciones del mapa. Al completarse un episodio, se reinician las posiciones de los jugadores y de las monedas aleatoriamente. Además, en cada paso de un episodio, los agentes observan un tensor de 4 canales que indica las posiciones absolutas de los jugadores (los dos primeros canales) y de las monedas (los dos segundos canales).

5.1.2. Simulación de dilemas sociales

Con el objetivo de comparar escenarios con y sin la presencia de un dilema social inherente, se definen dos configuraciones distintas para las recompensas asociadas a las acciones, i.e. dos posibles matrices de pagos. Estas configuraciones se describen mediante una triada de valores que representan, respectivamente:

$$[\text{recoger moneda propia}, \text{recoger moneda ajena}, \text{perder moneda propia}],$$

donde "perder" implica que el oponente recoge una moneda que no le corresponde. Así, las dos configuraciones consideradas son:

a) ***ND* (sin dilema)**: recoger una moneda propia otorga $+1$ punto; recoger una moneda ajena también otorga $+1$ punto; y que el oponente recoja una moneda propia (equivalente a perder una moneda propia) penaliza con -2 puntos. En este caso, la tripla correspondiente es $ND = [1, 1, -2]$.

b) ***PD* (dilema del prisionero)**: recoger una moneda propia otorga $+1$ punto; recoger una moneda ajena otorga $+2$ puntos; y perder una moneda propia implica una penalización de -3 puntos. Por tanto, la tripla resultante en este caso es $PD = [1, 2, -3]$.

Con estas dos configuraciones es posible simular dos escenarios iterados distintos. Cuando ambos agentes tienen la posibilidad de recoger monedas —ya sean propias o ajenas— de forma simultánea, el juego puede representarse mediante las matrices de recompensas correspondientes a cada configuración, presentadas en las tablas 5.1 y 5.2. Aquí, las acciones de cooperar (C) y traicionar (D) son simplemente recoger una moneda propia y recoger una moneda ajena.

Agente 1 \ Agente 2	Propia (C)	Ajena (D)
Propia (C)	$(1, 1)$	$(-1, 1)$
Ajena (D)	$(1, -1)$	$(-1, -1)$

Tabla 5.1: *Matriz de recompensas de la configuración* $ND = [1, 1, -2]$.

A partir de la tabla 5.1, se identifican los siguientes valores: $R = 1, T = 1, S = -1$ y $P = -1$. Estos satisfacen las condiciones $(1), (2), (3)$ de un dilema social, pero no cumplen la condición (4), ya que no se observa ni codicia ni miedo (véase la definición 3.1.2). En consecuencia, este caso no constituye un dilema social propiamente dicho. Además, presenta dos equilibrios de Nash: tanto la estrategia cooperativa (C, C) como la estrategia traidora (D, D) son estables.

Agente 1 \ Agente 2	Propia (C)	Ajena (D)
Propia (C)	$(1, 1)$	$(-2, 2)$
Ajena (D)	$(2, -2)$	$(-1, -1)$

Tabla 5.2: *Matriz de recompensas de la configuración* $PD = [1, 2, -3]$.

Por el contrario, en la tabla 5.2 se tiene que $R = 1, T = 2, S = -2$ y $P = -1$. Esta configuración cumple no solo las condiciones $(1), (2), (3)$, sino también la condición (4), ya que se manifiestan tanto codicia como miedo (véase la definición 3.1.2). Por tanto, esta situación corresponde a un dilema social análogo al dilema del prisionero clásico (véase la tabla 3.3), y presenta un único equilibrio de Nash: la estrategia traidora (D, D).

No obstante, ambas configuraciones son comparables en términos absolutos, ya que la suma de las recompensas obtenidas por ambos agentes es idéntica en cada una de las cuatro combinaciones posibles. Esto permite utilizar la puntuación conjunta —definida como la suma de las recompensas individuales— como métrica del rendimiento cooperativo de los agentes. Las puntuaciones totales en cada caso se presentan en la tabla 5.3.

Agente 1 \ Agente 2	Propia (C)	Ajena (D)
Propia (C)	2	0
Ajena (D)	0	-2

Tabla 5.3: *Puntuaciones totales de las configuraciones* $ND = [1, 1, -2]$ *y* $PD = [1, 2, -3]$.

5.1.3. Modelización de actitudes sociales

A diferencia de los trabajos que tienden a explorar casos extremos de cooperación o egoísmo, la presente propuesta introduce una parametrización continua de la actitud social de los agentes. Concretamente, se modifican las recompensas recibidas por cada agente mediante una combinación lineal de su propia recompensa y la del otro agente, según la fórmula:

$$r^i = \alpha^i \cdot r^i_{ex} + \beta^i \cdot r^{-i}_{ex}, \tag{5.1}$$

donde r^i_{ex} representa la recompensa pura (puntuación) obtenida por el agente y r^{-i}_{ex} la obtenida por su oponente. De esta forma, los coeficientes α y β definen la "actitud social" del agente, modelando cuánto valora su propio beneficio frente al del otro.

En este estudio se explora sistemáticamente el espacio de actitudes variando α y β en base a la restricción $\alpha^2 + \beta^2 = 1$, por lo que se mantiene constante el módulo de la recompensa general en todas las situaciones. Así, puede interpretarse esta exploración de actitudes como la descripción de la circunferencia de radio 1 (véase la figura 5.2) en el espacio de parámetros (α, β), siendo α y β las funciones trigonométricas del ángulo $\theta = \arctan\left(\frac{\alpha}{\beta}\right)$.

Explícitamente, lo que se hace es variar ángulos θ_1 y θ_2 —uno por agente— con incrementos de 45° en el rango $\theta^i \in [0°, 360°)$, luego se consideran los valores $\alpha^i := \cos(\theta^i)$ y $\beta^i := \sin(\theta^i)$, con $i \in \{1, 2\}$.

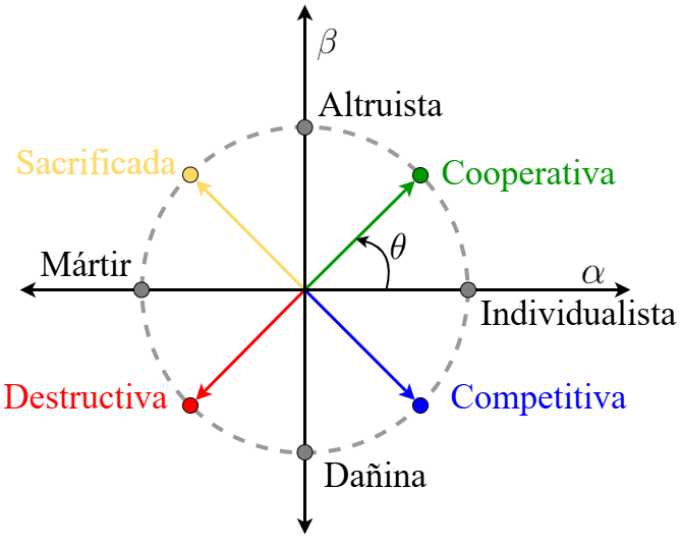


Figura 5.2: *Exploración sistemática del espacio de actitudes en el plano* (α, β) *bajo la restricción* $\alpha^2 + \beta^2 = 1$*, donde cada punto sobre la circunferencia representa una combinación de parámetros correspondiente a un ángulo* $\theta \in [0°, 360°)$*, variado en incrementos de* 45°*. Las componentes* $\alpha^i = \cos(\theta)$ *y* $\beta = \sin(\theta)$ *determinan la actitud específica del agente.*

5.2. Arquitectura del modelo

Este trabajo incluye dos versiones independientes del modelo *Actor-Critic*, adaptadas a diferentes marcos de implementación: una desarrollada desde cero utilizando JAX y `equinox`, y otra configurada a través de la librería `RLlib`, basada en PyTorch. Ambas versiones comparten el enfoque de aprendizaje basado en gradientes de política con estimación de ventajas —GAE, véase la ecuación (2.23)—, pero difieren en el nivel de control sobre la arquitectura y el flujo del entrenamiento. Mientras que la versión en JAX permite máxima flexibilidad y transparencia en el modelo, la versión en `RLlib` ofrece escalabilidad, paralelización automática y facilidad de integración con herramientas de monitoreo y evaluación. Esta doble implementación permite validar resultados y comprobar la consistencia del comportamiento aprendido bajo distintas plataformas.

5.2.1. Versión desarrollada en `JaxMARL`

La implementación manual en `JaxMARL`, de JAX, proporciona un control fino sobre todos los aspectos del modelo y del proceso de entrenamiento. El modelo actor-crítico se define explícitamente como una clase que contiene dos redes neuronales independientes (actor y crítico), ambas basadas en la clase `CustomMLP`. Esta clase permite construir perceptrones multicapa de forma modular, especificando manualmente el número de capas, sus tamaños y las inicializaciones de pesos (con escalado tipo Glorot para mejorar la estabilidad).

La arquitectura utilizada consta de tres capas ocultas con tamaños $(64, 64, 16)$, activación *tanh*, y salidas diferenciadas para acciones (actor) y valores (crítico). Las observaciones del entorno se aplanan antes de ser procesadas, lo que garantiza compatibilidad con redes densas. Esta estructura modular facilita tanto el análisis como la reutilización y extensión del modelo, y permite integrar de manera directa optimizadores funcionales como `optax`.

5.2.2. Versión desarrollada en `RLlib`

En paralelo, se ha desarrollado una versión equivalente utilizando el marco `RLlib`, de Ray, que permite escalar entrenamientos multiagente de forma eficiente. La arquitectura del modelo se especifica mediante el objeto `PPOConfig`, que permite definir no solo la topología de las redes, sino también todos los hiperparámetros del entrenamiento, siendo estos los propios de una estrategia de PPO estándar (véase la ecuación (2.38)).

En este caso, la red utilizada para el actor y el crítico está compuesta por tres capas ocultas de tamaños $(64, 64, 16)$ con activación *tanh* —equivalente al caso en JAX—. Se desactivan mecanismos como LSTM y atención, manteniendo un diseño sencillo y comparable al modelo en JAX, y la política de cada agente es definida explícitamente mediante un diccionario que asigna a cada agente su espacio de observación y acción correspondiente. Las políticas son entrenadas de forma independiente (aunque compartan configuración) bajo un esquema multiagente descentralizado.

5.3. Características del entrenamiento

Respecto a los algoritmos de entrenamiento empleados, Foerster et al. [15] implementan redes neuronales profundas, combinando capas convolucionales y recurrentes, y entrenan las políticas de ambos agentes simultáneamente mediante gradientes de política. En contraste, Lu et al. [35] emplean el algoritmo PPO. En este trabajo, se ha seguido un enfoque CTDE, entrenando de forma independiente una política (actor) para cada agente, pero con una función de valor (crítico) común. Además, el proceso de entrenamiento se basa en un *policy gradient*, concretamente mediante el algoritmo MAPPO (véase la ecuación (2.38)). A continuación, se describen los principales componentes del entrenamiento, tal y como se ha implementado con la librería `RLlib`:

1. **Inicialización del entorno y configuración**: El entorno *Coin Game* es adaptado a la interfaz de `RLlib` mediante la clase `CoopCoinsRLLibEnv`. La función `env_creator` permite crear instancias parametrizadas del entorno, incorporando aspectos como el tamaño del mapa, la matriz de pagos, los coeficientes de recompensa y el número de pasos por episodio. Este entorno es registrado mediante `register_env` para su uso dentro del ecosistema Ray.

2. **Definición de políticas multiagente**: Se configura un escenario multiagente con dos políticas independientes (*agent_0* y *agent_1*), asignadas explícitamente a sus respectivos agentes mediante una función de mapeo. Ambas políticas se entrenan simultáneamente pero de manera independiente, siguiendo un enfoque descentralizado.

3. **Selección de acciones**: Durante cada paso del entorno, los agentes seleccionan acciones a partir de su política, definida por una función *softmax* sobre los valores de salida. También se registran los valores estimados del estado y los *log*-probabilidades de las acciones tomadas, para su uso posterior en la función de pérdida.

4. **Función de pérdida**: El entrenamiento optimiza una combinación ponderada de tres términos:

   - Pérdida de política (*policy loss*): maximiza la probabilidad de acciones con ventajas positivas, con recorte (*clip*) controlado.
   - Pérdida de valor (*value loss*): minimiza el error cuadrático entre el valor predicho y el retorno observado.
   - Pérdida de entropía (*entropy loss*): incentiva la exploración al maximizar la entropía de la política, con un coeficiente de $\sigma = 0{,}05$.

5. **Estimación de ventajas**: Se utiliza GAE (véase la ecuación (2.23)) con $\gamma = 0{,}9$ y $\lambda = 0{,}95$ para calcular las ventajas, permitiendo una estimación más precisa del gradiente de política.

6. **Actualización de la política**: En cada episodio se recopilan trayectorias de longitud `NUM_INNER_STEPS` $= 150$, y se utilizan para entrenar la política durante `NUM_UPDATES` $= 4$ iteraciones.

7. **Ciclo de entrenamiento**: A lo largo de `NUM_EPOCHS` = 5000 episodio, los agentes interactúan con el entorno, acumulan datos, y actualizan sus políticas. Cada 100 episodios se guardan puntos de control (*checkpoints*).

8. **Persistencia de resultados**: Para cada ejecución, se crea una carpeta con marca temporal en el directorio de salida, donde se almacenan los modelos, métricas de entrenamiento (en formato CSV) y el archivo de configuración utilizado. Esto garantiza la trazabilidad y reproducibilidad del experimento.

De esta manera, el proceso de entrenamiento parte de un archivo de configuración externo, desde el cual se cargan los coeficientes de recompensa específicos (α y β) de cada agente, que permiten ponderar de forma diferenciada la recompensa individual y la recompensa social. Además, la lógica de entrenamiento también se parametriza externamente mediante los hiperparámetros presentados en la tabla 5.4.

Parámetro	**Valor**	**Descripción**
`NUM_ENVS`	4	Número de entornos simulados.
`NUM_INNER_STEPS`	150	Número de pasos por episodio.
`NUM_EPOCHS`	5000	Número de episodios entrenados.
`NUM_UPDATES`	4	Número de actualizaciones de la política por episodio.
`NUM_AGENTS`	2	Número de agentes entrenados de forma independiente.
γ	0,9	Factor de descuento.
λ	0,95	Factor de suavizado en GAE (ecuación (2.23)).
`CLIP_EPS`	0,2	Parámetro de recorte MAPPO (ecuación (2.38))
`VF_COEF`	0,5	Peso de la pérdida de valor.
`ENT_COEF`	0,05	Peso de la pérdida de entropía.
`GRID_SIZE`	3×3	Tamaño del mapa
`REWARD_COEF`	α y β	Modelización de la actitud social de cada agente.
`PAYOFF_MATRIX`	ND o PD	Matriz de pagos en función del dilema social.

Tabla 5.4: *Hiperparámetros principales del entrenamiento.*

5.4. Métricas de cooperación y desempeño

Para evaluar el desempeño y el comportamiento de los agentes en el entorno *CoopCoins*, se han definido y utilizado diversas métricas que permiten caracterizar tanto la calidad de la interacción como la efectividad de las estrategias aprendidas. Estas métricas se basan en los parámetros de actitud, las recompensas obtenidas y las acciones tomadas durante las ejecuciones. A continuación, se describen en detalle las principales métricas empleadas, junto con sus definiciones, utilidad, ventajas e inconvenientes.

Para definir estas métricas se debe especificar los datos que se han registrado durante la ejecución de las simulaciones:

- **Monedas propias recogidas**, M_p^i: número total de monedas capturadas por el agente i que le pertenecían, reflejando su rendimiento individual.
- **Monedas ajenas recogidas**, M_a^i: número total de monedas capturadas por el agente i que pertenecían al otro jugador, permitiendo evaluar posibles estrategias competitivas o de sabotaje.
- **Monedas propias obviadas**, O_p^i: monedas disponibles para el agente i y que le pertenecían pero decidió no recoger, lo que puede indicar decisiones tácticas.
- **Monedas ajenas obviadas**, O_a^i: monedas disponibles para el agente i que pertenecían al otro agente y que no fueron recogidas, lo que puede reflejar indicios de cooperación.
- **Sin monedas visibles**, T^i: instantes en los que el agente no tenía monedas accesibles.

Estos datos permiten un análisis más fino del comportamiento estratégico y de interacción. Por ejemplo, la comparación entre monedas propias y ajenas recogidas puede revelar tendencias hacia la cooperación o el egoísmo. Asimismo, los momentos sin monedas visibles ayudan a entender los patrones de exploración.

Recompensa

La *recompensa* —o recompensa extrínseca— del agente i frente al agente j, denotada como $r_{ex}^{i,j}$, cuantifica el rendimiento individual del agente según las monedas que recoge y el impacto que tiene la conducta del otro sobre su recompensa. Así, corresponde a la suma de monedas recogidas por dicho agente, aplicando sobre ella la matriz de pagos correspondiente:

$$r_{ex}^{i,j}(M_p^i, M_a^i, M_a^j) = \begin{cases} 1 \cdot M_p^i + 1 \cdot M_a^i - 2 \cdot M_a^j & \text{si } ND \\ 1 \cdot M_p^i + 2 \cdot M_a^i - 3 \cdot M_a^j & \text{si } PD \end{cases}. \tag{5.2}$$

Esta métrica refleja no solo la obtención de recursos por parte del agente, sino también las pérdidas que sufre si el otro jugador actúa en su contra, ofreciendo una medida clara de su beneficio neto.

Recompensa total

La *recompensa total* —o recompensa total extrínseca— de la pareja de agentes i y j, denotada como $R^{i,j}$, es simplemente la suma de sus recompensas individuales:

$$R^{i,j} = r_{ex}^{i,j} + r_{ex}^{j,i}. \tag{5.3}$$

Esta métrica permite evaluar el rendimiento colectivo, siendo especialmente útil para valorar el grado de cooperación o eficiencia global alcanzado por la pareja. A mayor valor de $R^{i,j}$, mayor es la productividad conjunta, independientemente de cómo se haya distribuido entre los dos agentes.

Coherencia

La *coherencia* del agente i frente al agente j, denotada como $c^{i,j}$, mide en qué medida la interacción entre ambos agentes resulta consistente con la actitud adoptada por i. Esta métrica combina la recompensa extrínseca individual con la del otro agente, ponderadas según los coeficientes de actitud de i —α^i para su propio beneficio y β^i para el beneficio ajeno—:

$$c^{i,j}(r^i_{ex}, r^j_{ex}) = \alpha^i \cdot r^i_{ex} + \beta^i \cdot r^j_{ex}. \tag{5.4}$$

Una coherencia alta indica que el resultado de la interacción está alineado con la orientación del agente, mientras que valores bajos reflejan desviaciones respecto a su actitud deseada. Esta métrica es clave para evaluar si las decisiones tomadas se corresponden con la política subyacente, aunque no garantiza altos niveles de recompensa ni éxito en términos extrínsecos.

Fuerza de la actitud

La *fuerza* de una actitud θ^i, denotada como $F(\theta^i)$, se define como la coherencia media que obtiene un agente i al adoptar dicha actitud, ponderando sobre el total de coherencias con el resto de agentes. Así, indica la capacidad de una actitud para mantenerse, independientemente de la actitud a la que se enfrenta.

$$F(\theta^i) = \frac{1}{N} \sum_{j \neq i} c^{i,j}. \tag{5.5}$$

Su principal ventaja es que resume el desempeño global en un único valor, facilitando la identificación de las actitudes más efectivas en términos generales. No obstante, esta agregación puede ocultar comportamientos inconsistentes o sensibles a ciertas condiciones específicas.

Resiliencia de la actitud

En contraste, la *resiliencia* de una actitud θ^i, denotada como $S(\theta^i)$, se define como el valor mínimo de la coherencia observado para un agente i con dicha actitud en el conjunto de configuraciones analizadas. Así, la resiliencia mide el peor caso: cuánto se desestabiliza una actitud en la peor de sus interacciones.

$$S(\theta^i) = \min_{j \neq i} c^{i,j}. \tag{5.6}$$

Es útil para identificar actitudes que, aunque quizás no sean las más fuertes en media, son más estables y menos vulnerables a situaciones desfavorables. A diferencia de la fuerza, que captura la efectividad general, la resiliencia se centra en la robustez. Ambas métricas son complementarias: una actitud fuerte pero no resiliente puede ser arriesgada, pues habrá casos muy extremos en la coherencia, mientras que una actitud resiliente pero no fuerte puede ser segura pero menos interesante.

Compatibilidad

La *compatibilidad* entre dos agentes i y j, denotada como $K^{i,j}$, mide el grado de alineación entre sus respectivas actitudes. Estas actitudes se representan mediante los coeficientes $\alpha^k = \cos(\theta^k)$ y $\beta^k = \sin(\theta^k)$, con $k \in \{i, j\}$ y θ^k es el ángulo que define la orientación actitudinal del agente k. La compatibilidad se calcula como:

$$K^{i,j} = \alpha^i \cdot \beta^j + \beta^i \cdot \alpha^j. \tag{5.7}$$

Esta métrica no evalúa el rendimiento ni implica una valoración de las actitudes en términos de éxito o coherencia, sino que describe hasta qué punto las motivaciones de ambos agentes son complementarias o contradictorias. Una compatibilidad alta sugiere que los términos cruzados —la preocupación del agente i por sí mismo combinada con la preocupación del agente j por el otro, y viceversa— están alineados, lo que puede facilitar la cooperación. En cambio, una compatibilidad baja (negativa) indica posibles tensiones o conflictos de interés.

La principal ventaja de esta medida es su sencillez y capacidad para capturar la afinidad actitudinal entre agentes de forma estática. No obstante, su interpretación debe hacerse con cautela: no refleja dinámicas temporales, ni garantiza un desempeño conjunto favorable, aunque sí ofrece una pista sobre el potencial de cooperación o conflicto latente entre las actitudes adoptadas.

Utilidad y limitaciones generales

La combinación de estas métricas proporciona un marco completo para evaluar tanto el desempeño cuantitativo como la naturaleza cualitativa de las estrategias desarrolladas por los agentes. Las métricas de fuerza, resiliencia y coherencia aportan información sobre la estabilidad actitudinal, mientras que la recompensa y los datos recogidos permiten medir resultados y analizar comportamientos emergentes.

No obstante, cada métrica tiene limitaciones: la coherencia no garantiza éxito en la tarea, la recompensa no considera dimensiones sociales, y la fuerza y la resiliencia solo son medidas relevantes si se considera la actitud. Por ello, se han utilizado estas métricas de forma complementaria, buscando convergencia en las conclusiones y realizando análisis contextuales para interpretar adecuadamente los resultados.

5.5. Planificación y fases del trabajo

El desarrollo del presente Trabajo de Fin de Máster se ha estructurado en varias fases claramente diferenciadas, que han permitido una evolución ordenada y progresiva del estudio. A continuación, se describen dichas fases, junto con el tiempo aproximado dedicado a cada una de ellas (véase la tabla 5.5).

Este desglose temporal refleja una aproximación iterativa e incremental, donde algunas fases (como el entrenamiento y análisis) se han solapado parcialmente para permitir una validación más ágil. La planificación ha servido también como guía para mantener el enfoque y evitar desviaciones respecto a los objetivos iniciales.

Fase	Descripción	Duración
Estudio preliminar	Revisión del estado del arte, bibliografía y análisis de entornos previos empleados en la literatura.	6 semanas
Diseño del entorno	Adaptación del entorno *Coin Game* para incorporar estructuras sociales, obteniendo así *CoopCoins*.	3 semanas
Diseño del modelo	Selección del algoritmo (MAPPO), definición de la arquitectura de red y elección de hiperparámetros.	1 semana
Implementación	Codificación del entorno adaptado, integración con `RLlib` y validación mediante pruebas unitarias y visuales.	3 semanas
Entrenamiento	Entrenamiento con distintas configuraciones, tuning de hiperparámetros y evaluación de la estabilidad.	2 semanas
Evaluación y métricas	Diseño e implementación de métricas cuantitativas sobre cooperación, equidad y recompensa.	2 semanas
Análisis de resultados	Interpretación de comportamientos emergentes, comparación entre entornos y validación de hipótesis.	1 semana
Redacción del TFM	Elaboración del documento final, organización de contenidos y generación de gráficos y tablas.	2 semanas
Total estimado		**20 semanas**

Tabla 5.5: *Fases del trabajo y tiempo estimado dedicado.*

CAPÍTULO 6

Resultados

En este capítulo se presentan los principales resultados obtenidos tras el entrenamiento de agentes en *CoopCoins* [23], una modificación del entorno *Coin Game* adaptada para estudiar dinámicas sociales en contextos cooperativos y competitivos. En este entorno simulado, dos agentes interactúan en un espacio compartido donde deben recoger monedas bajo distintos esquemas de recompensa. Esta adaptación permite incorporar explícitamente actitudes sociales diversas y modelar escenarios con y sin dilemas sociales. A través del análisis de métricas recogidas durante el entrenamiento, se busca comprender cómo influyen tanto las actitudes individuales como la estructura del entorno en la emergencia de distintos patrones de comportamiento colectivo.

Los resultados muestran que las actitudes sociales tienen un papel estructural en la dinámica de aprendizaje. A partir de métricas como la coherencia, la resiliencia, la fuerza o la compatibilidad, se caracteriza la estabilidad y expresividad de cada perfil actitudinal. Además, se observa que la coherencia del comportamiento aumenta cuando los agentes comparten objetivos alineados, lo que sugiere que ciertas actitudes tienden a generar estrategias más consistentes y eficaces. Estas propiedades permiten identificar qué actitudes favorecen la cooperación o la adaptación frente a diferentes configuraciones sociales, proporcionando una base cuantitativa para comparar su rendimiento y robustez.

Por otro lado, el análisis revela que la introducción de dilemas sociales, como el dilema del prisionero, tiene un efecto estructural profundo en la dinámica del entorno. Lejos de actuar como una perturbación externa, estos dilemas intensifican las tensiones entre intereses individuales y colectivos, amplificando las diferencias de comportamiento y recompensa según la actitud adoptada.

6.1. Efecto global del dilema del prisionero

Como punto de partida en el análisis de resultados, se examina el efecto global del dilema del prisionero sobre la recompensa total obtenida por los agentes. Para ello, se calcula la diferencia promedio entre las recompensas totales con y sin dilema, ponderada sobre todas las combinaciones de actitudes evaluadas. Los resultados indican que, en promedio, las configuraciones que incluyen el dilema conducen a un incremento significativo en la recompensa total (véase la figura 6.1), un hallazgo que, a primera vista, resulta llamativo.

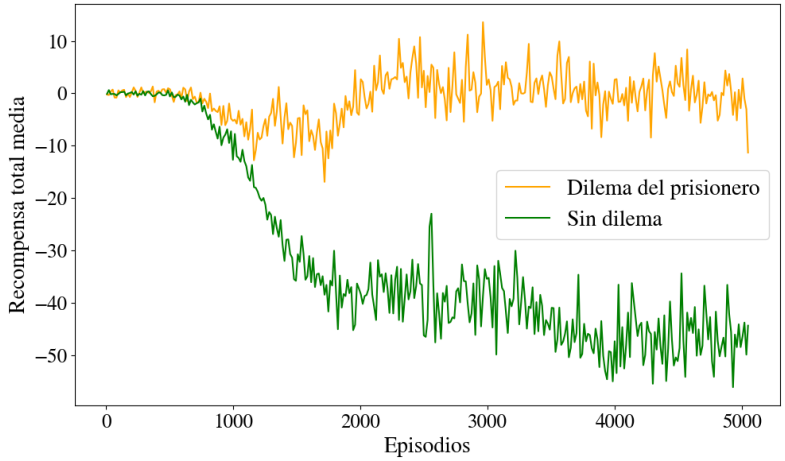
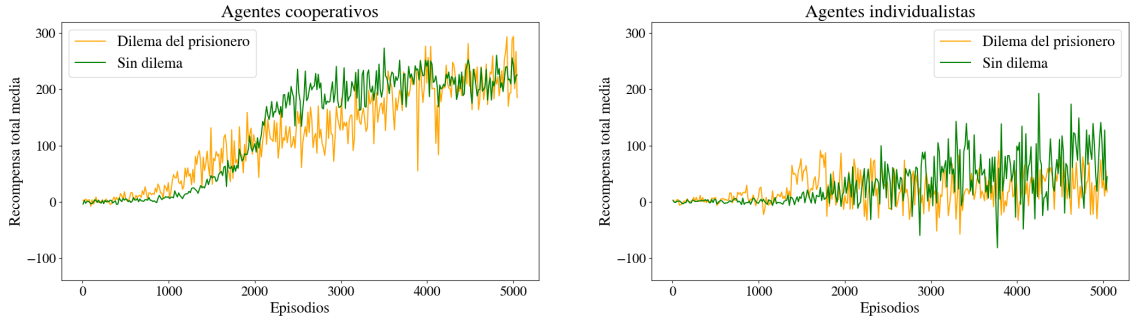


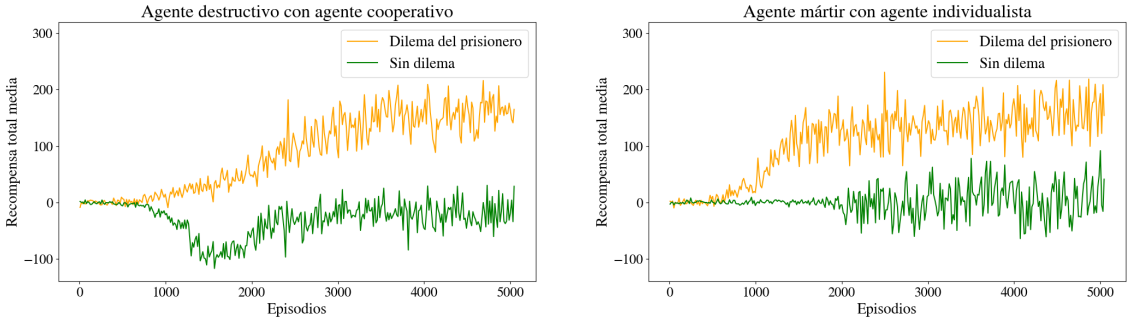
Figura 6.1: *Recompensa total media a lo largo del entrenamiento, comparando condiciones con y sin dilema del prisionero.*

Un análisis más detallado permite comprender que este resultado es, en realidad, esperable. En configuraciones donde las actitudes promueven comportamientos alineados —por ejemplo, dos agentes cooperativos (véase la figura 6.2a) o dos agentes individualistas (véase la figura 6.2b)—, la recompensa media es similar tanto con dilema como sin él. Sin embargo, en combinaciones de actitudes desalineadas —como la interacción entre un agente individualista y otro mártir (véase la figura 6.3a), o entre un agente destructivo y uno cooperativo (véase la figura 6.3b)—, la introducción del dilema permite que uno de los agentes se beneficie de forma desproporcionada, lo que eleva la recompensa media.



(a) *Actitudes alineadas cooperativamente.* (b) *Actitudes alineadas competitivamente.*

Figura 6.2: *Recompensa total en configuraciones con baja sensibilidad al dilema.*



(a) *Actitudes desalineadas mixtas.* (b) *Actitudes desalineadas puras.*

Figura 6.3: *Recompensa total en configuraciones con alta sensibilidad al dilema.*

Estos resultados reflejan que el impacto del dilema del prisionero no es homogéneo: su efecto depende en gran medida de la combinación de actitudes entre los agentes. En ciertos escenarios, su presencia apenas modifica la recompensa (véase la figura 6.2), mientras que en otros provoca aumentos marcados en el rendimiento conjunto (véase la figura 6.3).

6.1.1. Influencia de las actitudes sobre el efecto del dilema del prisionero

Con el objetivo de analizar cómo la interacción entre las actitudes de ambos agentes modula el impacto del dilema social, se ha construido una representación tridimensional del *efecto del dilema* para cada combinación de actitudes. Este efecto se define como la diferencia entre la recompensa total media obtenida en la condición con dilema y la obtenida sin dilema para una pareja específica de actitudes.

En la superficie mostrada en la figura 6.4, el eje X representa la actitud del primer agente (en grados), el eje Y la del segundo agente, y el eje Z el efecto del dilema correspondiente. Los valores positivos indican que la introducción del dilema mejora la recompensa conjunta, mientras que los valores negativos reflejan un deterioro del rendimiento total.

El análisis de la superficie revela una estructura altamente no lineal, con regiones claramente diferenciadas de ganancia y pérdida. Se observan máximos locales del efecto positivo del dilema en combinaciones asimétricas, particularmente cuando un agente adopta una actitud individualista o ligeramente cooperativa (0°, 45°) y el otro una actitud más competitiva o incluso destructiva (225°–315°). Las combinaciones $(0°, 315°)$ y $(45°, 270°)$ presentan incrementos superiores a +150 unidades en la recompensa total.

En contraste, el efecto del dilema es marcadamente negativo cuando ambos agentes adoptan actitudes altamente competitivas o destructivas. En particular, las combinaciones $(225°, 270°)$ y $(315°, 315°)$ registran caídas de hasta -115 y -110 respectivamente, lo que refleja una fuerte penalización a la coordinación entre agentes que persiguen objetivos abiertamente egoístas o dañinos.

Estos resultados sugieren que el dilema social no actúa de forma homogénea sobre el espacio de actitudes, sino que amplifica las dinámicas ya presentes entre los agentes. Para comprender mejor esta relación, resulta necesario analizar cómo se

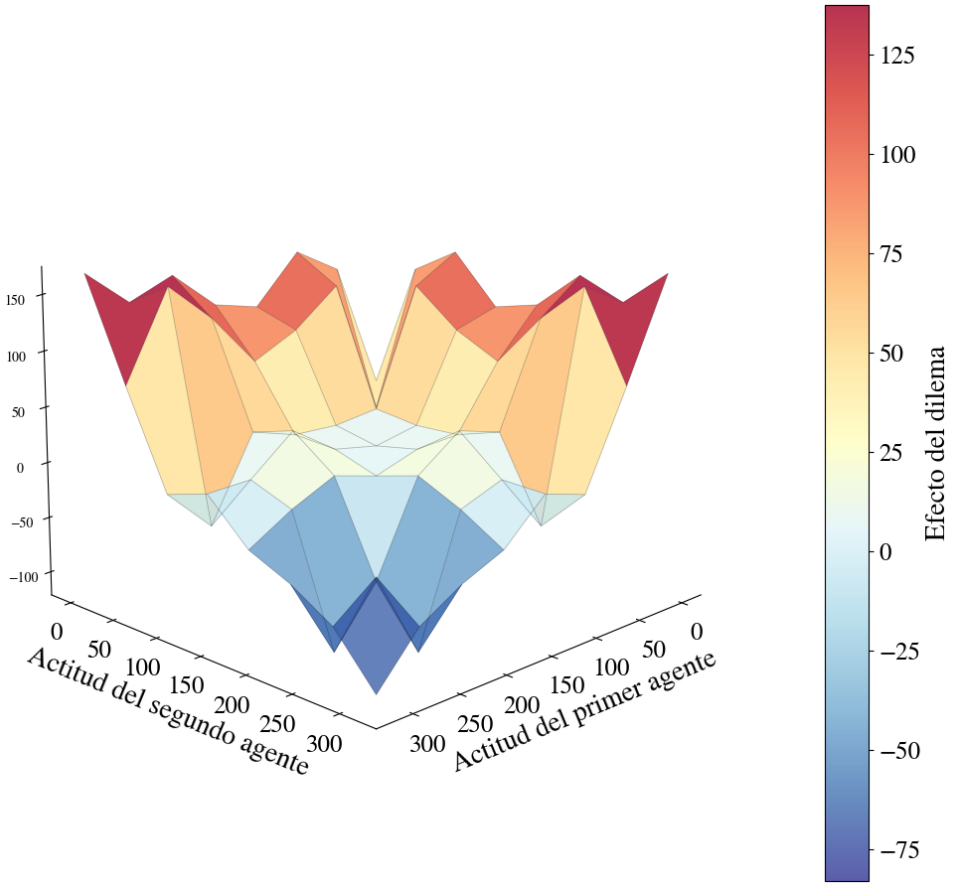


Figura 6.4: *Efecto del dilema social sobre la recompensa media conjunta, en función de las actitudes de los agentes.*

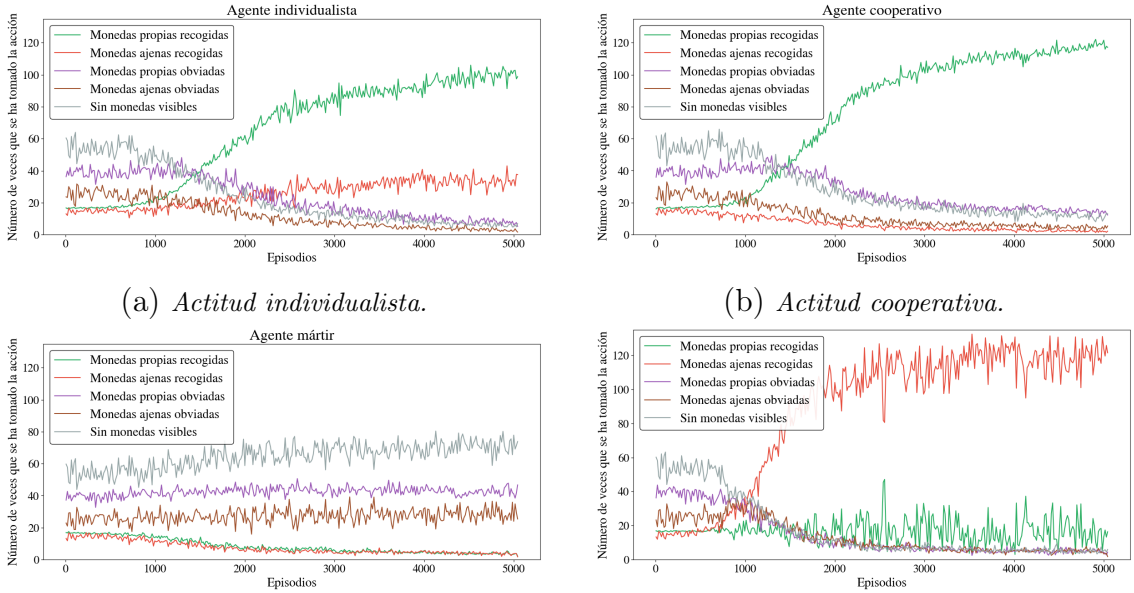
manifiestan las distintas actitudes en el comportamiento individual, independientemente del resultado global. El siguiente apartado se centra, por tanto, en la evolución de métricas asociadas a las decisiones de cada agente en función de su actitud.

6.2. Evolución de las actitudes individuales

En esta sección se analiza cómo varían las características del comportamiento de los agentes —es decir, las decisiones que toman— a lo largo del entrenamiento en función de sus actitudes individuales. Para cada configuración del entorno, ya sea con o sin dilema social, se seleccionan aquellas simulaciones en las que al menos uno de los agentes presenta una actitud determinada. A partir de estas, se calcula el promedio de las métricas correspondientes para todos los agentes que comparten dicha actitud. De este modo, es posible caracterizar el comportamiento asociado a cada actitud a partir de las decisiones observadas.

Las métricas analizadas corresponden a las registradas directamente durante el entrenamiento (véase la sección 5.4). Concretamente, se consideran las monedas propias y ajenas recogidas, las monedas propias y ajenas obviadas, y los episodios en los que no había monedas visibles. El análisis permite observar cómo ciertas características esperadas emergen en los distintos tipos de actitud:

- **Agente individualista** (0°, $\alpha = 1$ y $\beta = 0$): este agente tiende a recoger monedas, incluyendo ocasionalmente monedas ajenas, mientras que las decisiones de ignorar monedas disminuyen con el entrenamiento (véase la figura 6.5a).
- **Agente cooperativo** (45°, $\alpha = 0{,}707$ y $\beta = 0{,}707$): también recoge monedas con frecuencia, pero en este caso solamente las propias, para no perjudicar al otro jugador (véase la figura 6.5b).
- **Agente mártir** (180°, $\alpha = -1$ y $\beta = 0$): se caracteriza por evitar la recogida de monedas, lo que se refleja en una alta frecuencia de decisiones de rechazo y ausencia de monedas visibles, coherente con una actitud que interpreta cualquier ganancia como una pérdida (véase la figura 6.5c).
- **Agente destructivo** (225°, $\alpha = -0{,}707$ y $\beta = -0{,}707$): busca perjudicar al otro jugador, por lo que sus decisiones tienden a centrarse en recoger monedas ajenas (véase la figura 6.5d).



(a) *Actitud individualista.* (b) *Actitud cooperativa.*

(c) *Actitud mártir.* (d) *Actitud destructiva.*

Figura 6.5: *Evolución en las decisiones tomadas en función de la actitud del agente.*

En resumen, las decisiones observadas durante el entrenamiento reflejan de forma coherente la actitud individual de los agentes, y permiten caracterizarla con claridad. No obstante, resulta relevante considerar también la interacción entre agentes: aunque el comportamiento individual tiende a ser consistente con la actitud adoptada, este puede verse modulado por la actitud del otro agente, afectando así a la dinámica conjunta y a las decisiones finales.

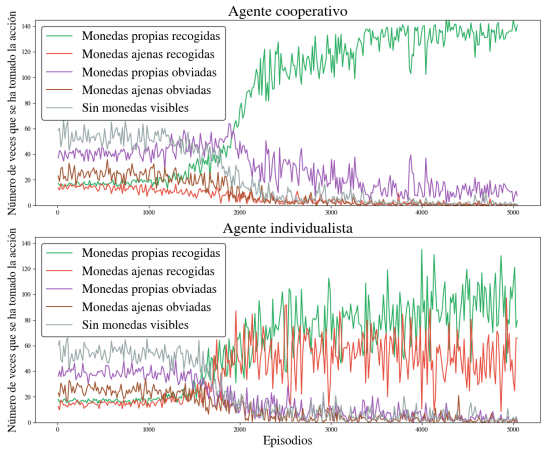
Esta constatación plantea una cuestión clave: ¿hasta qué punto el comportamiento de un agente depende no solo de su actitud propia, sino también de la del otro participante? Para responder a esta pregunta, es necesario ampliar el análisis y estudiar no solo actitudes aisladas, sino combinaciones concretas entre pares de agentes. Esta perspectiva permite explorar cómo emergen patrones de interacción específicos en función de la relación actitudinal establecida entre ambos.

6.3. Análisis de las combinaciones de actitudes

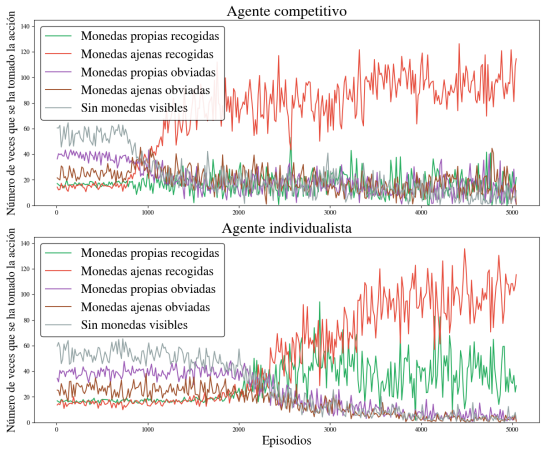
Además del estudio individual de cada actitud, resulta particularmente relevante analizar el rendimiento en función de combinaciones específicas de actitudes entre los dos agentes. Para ello, se considera cada pareja (θ^1, θ^2), o de forma equivalente, la cuádrupla $(\alpha^1, \beta^1, \alpha^2, \beta^2)$ que define completamente las actitudes de ambos agentes en una partida determinada.

Para cada combinación concreta se han generado gráficas que muestran en paralelo la evolución temporal de las métricas de ambos agentes. A diferencia del análisis previo, este enfoque permite observar explícitamente cómo el comportamiento inducido por una actitud individual varía en función de la actitud del oponente. Un caso especialmente ilustrativo es el de la actitud individualista, cuya expresión depende notablemente del tipo de interacción social a la que se enfrenta. Por ejemplo:

- **Influencia de actitudes cooperativas y competitivas sobre un agente individualista**: como se aprecia en las figuras 6.6a y 6.6b, un mismo agente individualista presenta comportamientos significativamente distintos según la actitud del otro agente. En presencia de un compañero cooperativo, tiende a priorizar la recogida de monedas propias sobre las ajenas (parte inferior de la figura 6.6a). En cambio, si el otro agente es competitivo, el individualista invierte esta preferencia y favorece la recogida de monedas ajenas (véase la figura 6.6b). Este patrón sugiere que el comportamiento del agente individualista no se rige únicamente por una motivación interna, sino que responde también a una influencia social derivada de la actitud del oponente.



(a) *Entrenamiento conjunto de un agente cooperativo con uno individualista.*



(b) *Entrenamiento conjunto de un agente competitivo con uno individualista.*

Figura 6.6: *Evolución en las decisiones tomadas durante el entrenamiento en función de la influencia ejercida por la actitud del otro agente. Caso sin dilema.*

Como se observa, algunas actitudes presentan una alta sensibilidad a la configuración del entorno social, modificando significativamente su comportamiento en función de la actitud del oponente. Este fenómeno sugiere que no todas las actitudes se expresan de forma estable, lo que motiva un análisis estructurado en torno a tres propiedades clave: coherencia, resiliencia y fuerza.

En primer lugar, desde la perspectiva del agente que toma decisiones, se introduce la noción de coherencia (véase la ecuación (5.4)). Esta medida cuantifica el grado en que el comportamiento observado es consistente con los parámetros internos de la actitud del agente, es decir, con sus valores (α, β), independientemente de la actitud del oponente. Un agente coherente actúa de manera fiel a su configuración, mostrando patrones estables y previsibles que reflejan su diseño actitudinal.

Desde la perspectiva de las interacciones entre agentes, resulta relevante distinguir dos propiedades complementarias que caracterizan el papel de una actitud en la dinámica social:

1. La resiliencia (véase la ecuación (5.6)) evalúa hasta qué punto una actitud mantiene su comportamiento frente a distintas actitudes del oponente, en el peor caso posible. Una actitud resiliente es resistente a la influencia externa en todos los casos, y muestra un patrón de decisión estable ante variaciones contextuales.

2. La fuerza (véase la ecuación (5.5)) mide la capacidad de una actitud para mantener su comportamiento en promedio. Una actitud fuerte permite al agente mantener una alta coherencia en casi todas las situaciones.

El análisis conjunto de estas tres propiedades proporciona una caracterización más profunda de las actitudes en juego, no solo desde el punto de vista individual, sino también en términos de su impacto relacional. Sin embargo, extraer estas propiedades directamente a partir de las métricas observadas durante el entrenamiento resulta poco operativo. Por ello, se han definido formalmente medidas específicas de coherencia, resiliencia y fuerza, que permiten comparar de forma cuantitativa y sistemática las distintas actitudes, facilitando su interpretación y revelando su papel funcional dentro del entorno.

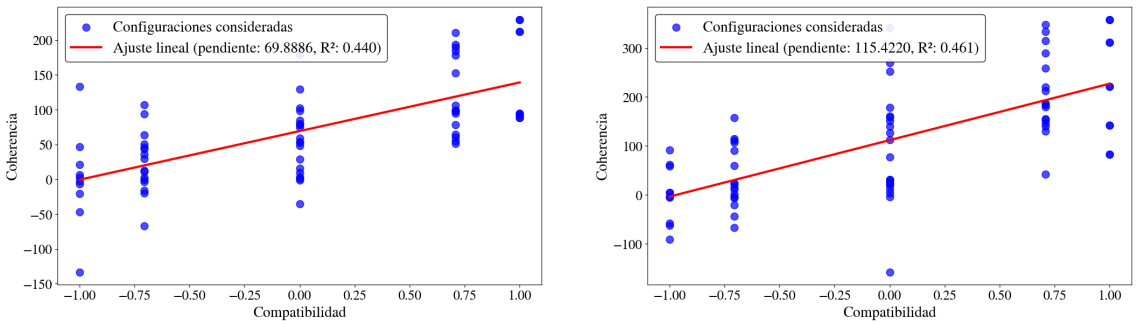
6.3.1. Compatibilidad entre actitudes y coherencia del comportamiento

Un aspecto particularmente relevante en la dinámica interagente es la relación entre la compatibilidad de las actitudes y la coherencia del comportamiento resultante. Intuitivamente, cuando las actitudes de ambos agentes están alineadas en sus objetivos, es más probable que cada uno actúe de forma coherente con su propia actitud. En cambio, cuando los objetivos subyacentes son divergentes o incluso opuestos, se incrementa la probabilidad de que uno de los agentes vea alterado su comportamiento por la influencia del otro, disminuyendo su coherencia interna.

Para formalizar esta idea, se introduce la noción de *compatibilidad* entre actitudes (véase la ecuación (5.7)). Esta métrica cuantifica el grado de alineamiento entre las direcciones actitudinales de los agentes: toma valores altos cuando ambos comparten objetivos complementarios (por ejemplo, competitividad y sacrificio), y valores bajos o negativos cuando sus actitudes son ortogonales o antagónicas.

El análisis se lleva a cabo evaluando, para cada combinación de actitudes, tanto la compatibilidad entre ellas como la coherencia individual de cada agente, diferenciando los escenarios con y sin dilema del prisionero. A partir de estos valores, se construyen diagramas de dispersión donde se representa la coherencia del agente en

función de la compatibilidad de las actitudes, y se ajusta una regresión lineal para identificar posibles tendencias.



(a) *Sin dilema.* (b) *Con dilema del prisionero.*

Figura 6.7: *Relación entre la coherencia del comportamiento y la compatibilidad entre actitudes, ajustada mediante regresión lineal.*

Tal y como se muestra en la figura 6.7, en ambos contextos se observa una correlación positiva clara: a mayor compatibilidad entre actitudes, mayor es la coherencia del comportamiento del agente. Este resultado pone de manifiesto que no solo las actitudes individuales influyen en la estrategia seguida, sino también su grado de alineamiento mutuo. Así, las configuraciones actitudinales más compatibles promueven comportamientos más consistentes y estables, lo que podría traducirse en mayor eficiencia o adaptabilidad dentro del entorno.

6.3.2. Fuerza y resiliencia de las actitudes

Una vez analizada la coherencia individual de los agentes, resulta especialmente interesante estudiar el efecto estructural de las distintas actitudes sobre el comportamiento observado. Para ello se introducen dos propiedades clave: la fuerza y la resiliencia de una actitud.

La fuerza de una actitud se define como el valor medio de coherencia que alcanza cuando es adoptada por un agente en distintos enfrentamientos. Representa, por tanto, la capacidad de una actitud para mantener un comportamiento estable y alineado con sus parámetros (α, β) en una amplia variedad de contextos. En cambio, la resiliencia evalúa el caso más adverso: es el valor mínimo de coherencia alcanzado por una actitud, es decir, su capacidad para evitar comportamientos marcadamente incoherentes incluso en condiciones hostiles. Una actitud resiliente no garantiza una coherencia alta constante, pero sí protege contra caídas extremas.

Ambas métricas se han calculado a partir de todos los episodios registrados, y se han construido sendas clasificaciones para comparar las actitudes en función de su fuerza y resiliencia (véase la figura 6.8).

Además, se ha estudiado la relación entre ambas propiedades para investigar si una actitud fuerte tiende también a ser resiliente. El coeficiente de correlación de Pearson obtenido ha sido de $r = 0{,}491$, lo que indica una correlación moderada positiva. Esto sugiere que, aunque ambas propiedades están relacionadas, no son equivalentes: existen actitudes que pueden mostrar una fuerza media alta pero ser vulnerables en situaciones concretas, y viceversa.

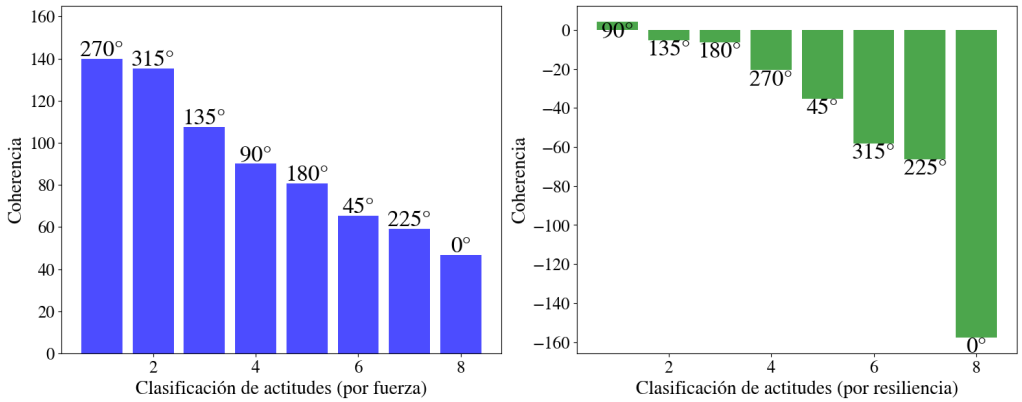


Figura 6.8: *Ranking de actitudes por fuerza (izquierda) y resiliencia (derecha).*

Un análisis más detallado de las tablas 6.1 y 6.2 revela observaciones notables. La actitud dañina (270°), centrada en perjudicar activamente al oponente, se posiciona como la más fuerte, i.e. su comportamiento es de los más estables a lo largo de distintos escenarios. Por su parte, la actitud altruista (90°), que prioriza el beneficio ajeno, resulta ser la más resiliente: incluso en los entornos más desfavorables, evita desviarse radicalmente de su propósito.

Ambas actitudes comparten una característica interesante: están fuertemente orientadas hacia el oponente, ya sea para ayudarlo (altruismo) o para perjudicarlo (daño). Esta direccionalidad parece dotarlas de una mayor consistencia interna, incluso frente a actitudes opuestas. Así, su foco externo las protege de las dinámicas internas del juego, actuando como "estrategias puras" en cualquier escenario.

En contraste, actitudes como la individualista (0°) o la destructiva (225°) se sitúan entre las más débiles y menos resilientes. La primera, centrada exclusivamente en el beneficio propio, parece carecer de la flexibilidad o adaptabilidad necesaria para sostener un comportamiento coherente cuando el entorno es adverso. La destructiva, que busca simultáneamente el propio beneficio y el perjuicio ajeno, presenta una motivación dual que probablemente genera conflictos internos en la toma de decisiones, debilitando su coherencia y su capacidad de adaptación.

Orden	Actitud	Ángulo	Fuerza
1	Dañina	270°	139,951
2	Competitiva	315°	135,180
3	Sacrificada	135°	107,511
4	Altruista	90°	90,167
5	Mártir	180°	80,630
6	Cooperativa	45°	65,315
7	Destructiva	225°	59,243
8	Individualista	0°	46,599

Tabla 6.1: *Ranking de actitudes según su fuerza (coherencia media).*

Estos resultados aportan una perspectiva valiosa para el diseño y análisis de sistemas multiagente: elegir actitudes fuertes y resilientes puede ayudar a garantizar una estrategia robusta y estable en entornos complejos o inciertos. Además, permiten

Rank	Actitud (α, β)	Ángulo	Resiliencia
1	Altruista	90°	4,249
2	Sacrificada	135°	−5,135
3	Mártir	180°	−6,385
4	Dañina	270°	−20,576
5	Cooperativa	45°	−35,297
6	Competitiva	315°	−58,079
7	Destructiva	225°	−66,305
8	Individualista	0°	−157,703

Tabla 6.2: *Ranking de actitudes según su resiliencia (mínima coherencia).*

identificar configuraciones vulnerables y potencialmente inestables que podrían ser mejoradas o sustituidas por estrategias más coherentes.

Resumen de resultados

La interacción entre las actitudes condiciona tanto la coherencia individual como la recompensa total, y modula el efecto del dilema. Para sintetizar las principales observaciones y facilitar su interpretación, se presenta a continuación un resumen visual de los hallazgos clave extraídos a lo largo del capítulo.

Aspecto analizado	Principales resultados
Efecto del dilema del prisionero	El dilema intensifica las dinámicas existentes entre actitudes: premia combinaciones asimétricas y penaliza interacciones simétricas destructivas. Su impacto no es homogéneo.
Comportamiento individual	Las decisiones de los agentes reflejan su actitud, pero también responden a la actitud del oponente, mostrando modulación social.
Compatibilidad actitudinal	A mayor compatibilidad entre actitudes, mayor coherencia del comportamiento individual: el alineamiento promueve estrategias estables.
Fuerza y resiliencia	Las actitudes orientadas al oponente, como la dañina o la altruista, presentan un comportamiento más resistente a la influencia externa.

Tabla 6.3: *Resumen de los resultados principales del capítulo.*

En conjunto, los resultados presentados evidencian el papel estructural que desempeñan las actitudes sociales en la dinámica de aprendizaje de sistemas multiagente. Desde el comportamiento individual hasta la interacción entre perfiles y su respuesta ante dilemas sociales, se ha mostrado cómo la configuración actitudinal condiciona de forma profunda tanto la coherencia de los agentes como su capacidad de adaptación. Estas observaciones no solo permiten clasificar actitudes según criterios de estabilidad y resiliencia, sino que ofrecen una base cuantitativa para el diseño de agentes estratégicamente complementarios.

CAPÍTULO 7

Conclusiones

Las sociedades complejas, tanto humanas como artificiales, se sostienen sobre una paradoja fundamental: cada agente que las compone es autónomo, libre en sus decisiones, pero está a la vez envuelto en una red de interdependencias. ¿Cómo es posible, entonces, que emerjan estructuras sociales estables en sistemas poblados por entidades artificiales? Esta era la pregunta que motivaba este Trabajo de Fin de Máster, y sobre ella se ha tratado de arrojar luz desde el marco teórico del Aprendizaje por Refuerzo Multiagente (*Multi-Agent Reinforcement Learning*, MARL) y la teoría de juegos.

Ecosistemas de actitudes

En el corazón de este estudio late una hipótesis tan sencilla como ambiciosa: es posible modular el comportamiento emergente de agentes inteligentes no solo a través de incentivos, sino mediante *actitudes* —modos de relacionarse con los demás que no dictan qué acción tomar, pero sí qué tipo de estrategia aprender—. Estas actitudes, como el individualismo, la cooperación o la destructividad, funcionan como lentes a través de las cuales los agentes interpretan su entorno social. No son comportamientos en sí mismos, sino predisposiciones que alteran el aprendizaje.

A lo largo del trabajo se ha demostrado que estas actitudes no son decorativas ni anecdóticas. Por el contrario, han mostrado una notable capacidad de estructurar el aprendizaje: los agentes individualistas optimizan su propio beneficio a corto plazo; los cooperativos exploran estrategias que favorecen el rendimiento conjunto; los destructivos sabotean incluso a costa de sí mismos. Pero lo más revelador es que estas actitudes no operan en el vacío: su manifestación depende profundamente del entorno relacional, es decir, de con quién interactúan.

Esta dependencia mutua introduce una dimensión de **influencia social** en el aprendizaje, que rara vez es atendida en los entornos MARL clásicos. Lo que se observa, en términos prácticos, es que un mismo agente puede comportarse de forma cooperativa, egoísta o incluso ambigua dependiendo de la actitud de su oponente. No se trata de inconsistencia, sino de adaptación estratégica: los agentes, incluso cuando están sesgados por una actitud interna, aprenden a leer el contexto social para maximizar sus beneficios según ese sesgo.

Coherencia, resiliencia, fuerza: más allá del rendimiento

Un logro importante de este trabajo ha sido el desarrollo de métricas actitudinales que permiten ir más allá de la mera recompensa acumulada para analizar la consistencia y estabilidad de las políticas aprendidas. Se ha propuesto una taxonomía tridimensional —coherencia, resiliencia y fuerza— que permite clasificar las actitudes no solo por su rendimiento, sino por su capacidad de mantener un patrón de comportamiento frente a influencias externas.

- **La coherencia** mide cuán predecible es un agente dado su sesgo actitudinal. Un agente coherente es fiel a su actitud, no se traiciona a sí mismo.
- **La resiliencia** indica la resistencia frente a la influencia del otro. Es decir, cuánto varía el comportamiento del agente en el peor caso, cuando el entorno social se vuelve hostil, contradictorio o cooperativo en exceso.
- **La fuerza** es la robustez global de una actitud frente a la interacción social. Una actitud fuerte es aquella que, en general, no se ve influenciada por las otras actitudes.

Gracias a estas métricas, se ha podido responder con precisión a preguntas que antes eran cualitativas: ¿qué actitudes son estructuralmente sólidas? ¿cuáles ceden fácilmente al chantaje cooperativo o al sabotaje? ¿cómo se comportan los agentes frente a oponentes impredecibles?

Compatibilidad y simbiosis actitudinal

Otra línea fructífera del trabajo ha sido el análisis de la compatibilidad entre actitudes. Aquí se ha demostrado, con datos empíricos, algo que la teoría de juegos sugería pero que rara vez se explora con tal claridad: la alineación actitudinal promueve comportamientos más estables y cooperativos. Dos agentes cooperativos, por ejemplo, convergen fácilmente hacia políticas de reparto eficiente. Pero incluso un agente competitivo y uno sacrificado pueden generar una forma de simbiosis: el primero ataca, el segundo cede, y ambos estabilizan su comportamiento en un equilibrio disfuncional pero estable.

Este hallazgo tiene implicaciones importantes para el diseño de sistemas multiagente. No basta con desarrollar agentes "inteligentes"; es crucial que sus sesgos sociales sean complementarios, y que el diseñador entienda la dinámica que surge de sus combinaciones. En este sentido, se sugiere que el diseño actitudinal puede ser una herramienta tan potente como la ingeniería de recompensas.

El dilema como catalizador

La introducción de dilemas sociales, como el clásico dilema del prisionero, añade una capa adicional de complejidad. El dilema no afecta a todos los agentes por igual, ni de forma lineal: su impacto está fuertemente modulado por la configuración actitudinal del sistema. Quizá lo más sorprendente ha sido observar que el dilema no siempre es negativo. En determinadas configuraciones, puede ser incluso beneficioso: una tensión que estabiliza relaciones. Este efecto depende de cómo las actitudes interactúan en ese terreno común de incertidumbre, riesgo y posible traición.

Un camino abierto

Este trabajo no cierra una línea de investigación: las preguntas que emergen de sus resultados son tan ricas como las que intentaba responder. ¿Qué ocurre cuando el número de agentes crece? ¿Cómo se transforma la dinámica actitudinal en redes complejas, donde cada agente interactúa con muchos otros? ¿Qué papel juega la comunicación explícita, la memoria o la reputación en estos contextos? ¿Cómo diseñar entornos artificiales donde convivan, cooperen y compitan agentes con diferentes objetivos y actitudes?

La respuesta a estas preguntas no es solo académica. En un entorno donde algoritmos autónomos ajustan precios, planifican rutas, asignan recursos y toman decisiones coordinadas entre sí, comprender las reglas emergentes de su interacción se vuelve una necesidad urgente. Este trabajo ha demostrado que es posible modelar, medir y modular actitudes artificiales, y que hacerlo permite no solo mejorar el rendimiento, sino entender mejor **cómo surge lo social desde lo individual**.

Y quizá esa sea la verdadera conclusión de este proyecto: que el comportamiento colectivo no es una suma de inteligencias, sino una forma emergente de orden, nacida del conflicto, la adaptación y la posibilidad de cambio. No hay cooperación sin riesgo, ni estrategia sin el otro. En un mundo artificial cada vez más poblado, comprender esto no es un lujo teórico, sino una necesidad profundamente humana.

Bibliografía

[1] J. J. Aloor, S. N. Nayak, S. Dolan, and H. Balakrishnan. Cooperation and fairness in multi-agent reinforcement learning. *ACM Journal on Autonomous Transportation Systems*, 2(2):1–25, Dec. 2024. ISSN 2833-0528. doi: 10.1145/3702012. URL http://dx.doi.org/10.1145/3702012.

[2] R. Axelrod. *The Evolution of Cooperation*. Basic, New York, 1984.

[3] B. Banerjee, J. Lyle, L. Kraemer, and R. Yellamraju. Sample bounded distributed reinforcement learning for decentralized pomdps. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2012.

[4] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(5):834–846, 1983.

[5] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[6] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1996.

[7] N. Brown and T. Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890, 2019. doi: 10.1126/science.aay2400. URL https://www.science.org/doi/abs/10.1126/science.aay2400.

[8] P.-A. Chen and D. Kempe. Altruism, selfishness, and spite in traffic routing. In *Proceedings of the 9th ACM Conference on Electronic Commerce*, EC '08, page 140–149, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581699. doi: 10.1145/1386790.1386816. URL https://doi.org/10.1145/1386790.1386816.

[9] C. Claus and C. Boutilier. The dynamics of learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI)*, pages 746–752. AAAI Press, 1998.

[10] D. de Kok. Robocup 2013, eindhoven, 2013. URL https://commons.wikimedia.org/wiki/File:13-06-28-robocup-eindhoven-005.jpg. Licensed under CC BY-SA 3.0.

[11] Y. Du, J. Z. Leibo, U. Islam, R. Willis, and P. Sunehag. A review of cooperation in multi-agent learning, 2023. URL https://arxiv.org/abs/2312.05162.

[12] E. Fehr and S. Gächter. Altruistic punishment in humans. *Nature*, 415(6868): 137–140, 2002.

[13] E. Fehr and K. M. Schmidt. A theory of fairness, competition, and cooperation. *The Quarterly Journal of Economics*, 114(3):817–868, 1999. ISSN 00335533, 15314650. URL http://www.jstor.org/stable/2586885.

[14] P. Feng, J. Liang, S. Wang, X. Yu, X. Ji, Y. Chen, K. Zhang, R. Shi, and W. Wu. Hierarchical consensus-based multi-agent reinforcement learning for multi-robot cooperation tasks, 2024. URL https://arxiv.org/abs/2407.08164.

[15] J. N. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch. Learning with opponent-learning awareness. *CoRR*, abs/1709.04326, 2017. URL http://arxiv.org/abs/1709.04326.

[16] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. *CoRR*, abs/1705.08926, 2017. URL http://arxiv.org/abs/1705.08926.

[17] E. A. Hansen, D. S. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, page 709–715. AAAI Press, 2004. ISBN 0262511835.

[18] G. Hardin. The tragedy of the commons. *Science*, 162(3859):1243–1248, 1968.

[19] M. J. Hausknecht and P. Stone. Deep recurrent q-learning for partially observable mdps. *CoRR*, abs/1507.06527, 2015. URL http://arxiv.org/abs/1507.06527.

[20] E. Hughes, J. Z. Leibo, M. Phillips, K. Tuyls, E. Dueñez Guzman, A. García Castañeda, I. Dunning, T. Zhu, K. McKee, R. Koster, H. Roff, and T. Graepel. Inequity aversion improves cooperation in intertemporal social dilemmas. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/7fea637fd6d02b8f0adf6f7dc36aed93-Paper.pdf.

[21] A. A. Hussain, F. Belardinelli, and G. Piliouras. Asymptotic convergence and performance of multi-agent q-learning dynamics. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2023.

[22] M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson. Deep variational reinforcement learning for pomdps, 2018. URL https://arxiv.org/abs/1806.02426.

[23] S. L. Iglesias. Coopcoins: Adapted marl environment of the coin game and training framework. https://github.com/samuellozanoiglesias/CoopCoins, 2025. Accessed: 2025-06-20.

[24] N. Jaques, A. Lazaridou, E. Hughes, Ç. Gülçehre, P. A. Ortega, D. Strouse, J. Z. Leibo, and N. de Freitas. Intrinsic social motivation via causal influence in multi-agent RL. *CoRR*, abs/1810.08647, 2018. URL http://arxiv.org/abs/1810.08647.

[25] F. Kong, Y. Huang, S.-C. Zhu, S. Qi, and X. Feng. Learning to balance altruism and self-interest based on empathy in mixed-motive games, 2025. URL https://arxiv.org/abs/2410.07863.

[26] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. URL http://www.jstor.org/stable/2236703.

[27] A. Labiosa and J. P. Hanna. Multi-robot collaboration through reinforcement learning and abstract simulation, 2025. URL https://arxiv.org/abs/2503.05092.

[28] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel. Multi-agent reinforcement learning in sequential social dilemmas, 2017. URL https://arxiv.org/abs/1702.03037.

[29] A. Lerer and A. Peysakhovich. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *CoRR*, abs/1707.01068, 2017. URL http://arxiv.org/abs/1707.01068.

[30] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica. Rllib: Abstractions for distributed reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3053–3062, 2018.

[31] L.-J. Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992. doi: 10.1007/BF00992699. URL https://doi.org/10.1007/BF00992699.

[32] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In W. W. Cohen and H. Hirsh, editors, *Machine Learning Proceedings 1994*, pages 157–163. Morgan Kaufmann, San Francisco (CA), 1994. ISBN 978-1-55860-335-6. doi: https://doi.org/10.1016/B978-1-55860-335-6.50027-1. URL https://www.sciencedirect.com/science/article/pii/B9781558603356500271.

[33] H. Liu, Y. Li, Z. S. Wu, and K.-W. Chang. Policy gradient methods for deep reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 12(1):1–98, 2019.

[34] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *CoRR*, abs/1706.02275, 2017. URL http://arxiv.org/abs/1706.02275.

[35] C. Lu, T. Willi, C. S. de Witt, and J. Foerster. Model-free opponent shaping, 2022. URL https://arxiv.org/abs/2205.01447.

[36] X. Lyu, A. Baisero, Y. Xiao, B. Daley, and C. Amato. On centralized critics in multi-agent reinforcement learning, 2024. URL https://arxiv.org/abs/2408.14597.

[37] M. W. Macy and A. Flache. Learning dynamics in social dilemmas. *Proceedings of the National Academy of Sciences*, 99(suppl_3):7229–7236, 2002. doi: 10.1073/pnas.092080099. URL https://www.pnas.org/doi/abs/10.1073/pnas.092080099.

[38] K. R. McKee, I. Gemp, B. McWilliams, E. A. Duèñez Guzmán, E. Hughes, and J. Z. Leibo. Social diversity and social preferences in mixed-motive reinforcement learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, page 869–877, Richland, SC, 2020. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

[39] K. R. McKee, E. Hughes, T. O. Zhu, M. J. Chadwick, R. Koster, A. G. Castaneda, C. Beattie, T. Graepel, M. Botvinick, and J. Z. Leibo. A multi-agent reinforcement learning model of reputation and cooperation in human groups, 2023. URL https://arxiv.org/abs/2103.04982.

[40] F. S. Melo. Convergence of Q-learning: A simple proof. Technical report, Institute of Systems and Robotics, 2001.

[41] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

[42] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. doi: 10.1038/nature14236.

[43] J. F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

[44] A. Y. Ng, D. Harada, and S. J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, page 278–287, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1558606122.

[45] Z. Ning and L. Xie. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2):73–91, 2024. ISSN 2949-8554. doi: https://doi.org/10.1016/j.jai.2024.02.003. URL https://www.sciencedirect.com/science/article/pii/S2949855424000042.

[46] F. A. Oliehoek and C. Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

[47] D. Radke, K. Larson, and T. Brecht. The importance of credo in multiagent learning, 2023. URL https://arxiv.org/abs/2204.07471.

[48] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning, 2018. URL https://arxiv.org/abs/1803.11485.

[49] J.-J. Rousseau. *Discours sur l'origine et les fondements de l'inégalité parmi les hommes*. Marc-Michel Rey, Amsterdam, 1755. Discourse on the Origin of Inequality.

[50] G. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. *Technical Report CUED/F-INFENG/TR 166*, 11 1994.

[51] A. Rutherford, B. Ellis, M. Gallici, J. Cook, A. Lupu, G. Ingvarsson, T. Willi, R. Hammond, A. Khan, C. S. de Witt, A. Souly, S. Bandyopadhyay, M. Samvelyan, M. Jiang, R. T. Lange, S. Whiteson, B. Lacerda, N. Hawes, T. Rocktäschel, C. Lu, and J. N. Foerster. Jaxmarl: Multi-agent rl environments and algorithms in jax. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[52] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. J. Rudner, C.-M. Hung, P. H. S. Torr, J. Foerster, and S. Whiteson. The starcraft multi-agent challenge, 2019. URL https://arxiv.org/abs/1902.04043.

[53] T. C. Schelling. *The Strategy of Conflict*. Harvard University Press, 1960.

[54] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[55] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL https://arxiv.org/abs/1506.02438.

[56] S. Shalev-Shwartz, S. Shammah, and A. Shashua. Safe, multi-agent, reinforcement learning for autonomous driving, 2016. URL https://arxiv.org/abs/1610.03295.

[57] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953.

[58] J. Smit and F. P. Santos. Fairness and cooperation between independent reinforcement learners through indirect reciprocity. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '24, page 2468–2470, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400704864.

[59] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017. URL https://arxiv.org/abs/1706.05296.

[60] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1999.

[61] M. Tan. Multi-agent reinforcement learning: independent versus cooperative agents. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning*, ICML'93, page 330–337, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. ISBN 1558603077.

[62] A. Tucker. A two-person dilemma, 1950. Unpublished notes, Stanford, May 1950. Reprinted in Rasmussen, Eric, Ed., Readings in Games and Information, Malden, MA: Blackwell, pp. 7–8.

[63] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. P. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350 – 354, 2019. URL https://api.semanticscholar.org/CorpusID:204972004.

[64] A. Viseras, M. Meissner, and J. Marchal. Wildfire front monitoring with multiple uavs using deep q-learning. *IEEE Access*, pages 1–1, 2021. doi: 10.1109/ACCESS.2021.3055651.

[65] J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, 1947.

[66] J. Wang, Z. Jiang, and Z. S. Wu. On the convergence of actor-critic methods with deep learning. *Journal of Machine Learning Research*, 21(108):1–28, 2020.

[67] C. J. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, University of Cambridge, 1989.

[68] C. J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[69] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[70] V. Xiang, L. Cross, J.-P. Fränken, and N. Haber. From centralized to self-supervised: Pursuing realistic multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2312.08662.

[71] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. M. Bayen, and Y. Wu. The surprising effectiveness of MAPPO in cooperative, multi-agent games. *CoRR*, abs/2103.01955, 2021. URL https://arxiv.org/abs/2103.01955.

[72] C. Yu, X. Yang, J. Gao, J. Chen, Y. Li, J. Liu, Y. Xiang, R. Huang, H. Yang, Y. Wu, and Y. Wang. Asynchronous multi-agent reinforcement learning for efficient real-time multi-robot cooperative exploration, 2023. URL https://arxiv.org/abs/2301.03398.

[73] L. Yuan, Z. Zhang, L. Li, C. Guan, and Y. Yu. A survey of progress on cooperative multi-agent reinforcement learning in open environment, 2023. URL https://arxiv.org/abs/2312.01058.

[74] G. Zhang, W. Hu, D. Cao, Z. Zhang, Q. Huang, Z. Chen, and F. Blaabjerg. A multi-agent deep reinforcement learning approach enabled distributed energy management schedule for the coordinate control of multi-energy hub with gas, electricity, and freshwater. *Energy Conversion and Management*, 255:115340, 2022. ISSN 0196-8904. doi: https://doi.org/10.1016/j.enconman.2022.115340. URL https://www.sciencedirect.com/science/article/pii/S0196890422001364.

[75] K. Zhang, Z. Yang, and T. Başar. Fully decentralized multi-agent reinforcement learning with networked agents. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 5867–5876, 2018.

[76] K. Zhang, Z. Yang, H. Xu, and T. Başar. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[77] Z. Zhang, C. Zhang, N. Liu, S. Qi, Z. Rong, S.-C. Zhu, S. Cui, and Y. Yang. Heterogeneous value alignment evaluation for large language models, 2024. URL https://arxiv.org/abs/2305.17147.

CAPÍTULO 8

Apéndices

En este capítulo auxiliar se incluyen contenidos no necesarios para el correcto entendimiento del documento, pero sí muy aconsejables para su profundización y completo estudio. Los apéndices se dividen en dos secciones: un primer apéndice A con demostraciones de resultados presentados en el texto y un segundo apéndice B con el código empleado en el trabajo.

8.1. Apéndice A: Demostraciones

El *policy gradient theorem*, teorema 2.1.5, enunciado en la sección 2.1.2, se demuestra como sigue:

Teorema 8.1.1. *Para cualquier proceso de decisión de Markov con las condiciones descritas en la definición 2.1.1, siendo la política* $\pi_{\boldsymbol{\theta}} : \mathcal{A} \times \mathcal{S} \to [0, 1]$*, dependiente de unos parámetros* $\boldsymbol{\theta} \in \mathbb{R}^d$ *con* $d < |\mathcal{S}|$*, y considerando la distribución de estados de la política* $\mu^{\pi_{\boldsymbol{\theta}}}(s)$ *dada por la definición 2.1.4. Entonces, si la política es diferenciable con respecto a dichos parámetros, el gradiente de la recompensa acumulada* $J(\theta)$*, dada por la ecuación* (2.12)*, es:*

$$\nabla J(\boldsymbol{\theta}) \propto \sum_{s \in \mathcal{S}} \mu^{\pi_{\boldsymbol{\theta}}}(s) \sum_{a \in \mathcal{A}} \nabla \pi_{\boldsymbol{\theta}}(s, a) \cdot Q^{\pi_{\boldsymbol{\theta}}}(s, a), \tag{8.1}$$

donde $Q^{\pi_\theta}(s, a)$ *es la función de acción-valor.*

Demostración. (Adaptación de [60]). Considerando, en primer lugar, las funciones de valor (2.4) y de acción-valor (2.5) para un estado $s \in \mathcal{S}$ cualquiera. Estas funciones

se pueden relacionar mediante la política de la forma (2.6):

$$V^{\pi_{\boldsymbol{\theta}}}(s) = \sum_{a \in \mathcal{A}} \pi_{\boldsymbol{\theta}}(s,a) \cdot Q^{\pi_{\boldsymbol{\theta}}}(s,a).$$

Con esto, se puede derivar a ambos lados con respecto a los parámetros $\boldsymbol{\theta}$. Para aligerar la notación se asumirá que todas las derivadas son con respecto a $\boldsymbol{\theta}$:

$$\begin{aligned}\nabla V^{\pi_{\boldsymbol{\theta}}}(s) &= \nabla \sum_{a \in \mathcal{A}} \pi_{\boldsymbol{\theta}}(s,a) \cdot Q^{\pi_{\boldsymbol{\theta}}}(s,a) = \\ &= \sum_{a \in \mathcal{A}} \left[\nabla \pi_{\boldsymbol{\theta}}(s,a) \cdot Q^{\pi_{\boldsymbol{\theta}}}(s,a) + \pi_{\boldsymbol{\theta}}(s,a) \cdot \nabla Q^{\pi_{\boldsymbol{\theta}}}(s,a)\right].\end{aligned}$$

Entonces, se considera la relación (2.7) para la función de acción-valor. Para aligerar aún más la notación, se quitará la dependencia explícita de π con respecto a $\boldsymbol{\theta}$, así como la de Q y V con respecto a la política π:

$$\nabla V(s) = \sum_{a \in \mathcal{A}} \left[\nabla \pi(s,a) \cdot Q(s,a) + \pi(s,a) \cdot \nabla \left(r(s,a) + \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s,a) \cdot V(s')\right)\right].$$

Como la recompensa $r(s,a)$ no depende de los parámetros $\boldsymbol{\theta}$, pues no depende de la política, se puede cancelar su derivada y volver a utilizar el mismo argumento de la ecuaciones (2.6) y de la ecuación (2.7):

$$\begin{aligned}\nabla V(s) &= \sum_{a \in \mathcal{A}} \left[\nabla \pi(s,a) \cdot Q(s,a) + \pi(s,a) \cdot \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s,a) \cdot \nabla V(s')\right] = \\ &= \sum_{a \in \mathcal{A}} \left[\nabla \pi(s,a) \cdot Q(s,a) + \pi(s,a) \cdot \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s,a) \cdot \right. \\ &\qquad \left. \cdot \nabla \left(\sum_{a' \in \mathcal{A}} \pi(s',a') \cdot Q(s',a')\right)\right] = \\ &= \sum_{a \in \mathcal{A}} \left[\nabla \pi(s,a) \cdot Q(s,a) + \pi(s,a) \cdot \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s,a) \cdot \right. \\ &\qquad \left. \cdot \sum_{a' \in \mathcal{A}} \left[\nabla \pi(s',a') \cdot Q(s',a') + \pi(s',a') \cdot \nabla Q(s',a')\right]\right] = \\ &= \sum_{a \in \mathcal{A}} \left[\nabla \pi(s,a) \cdot Q(s,a) + \pi(s,a) \cdot \sum_{s' \in \mathcal{S}} \gamma \cdot p(s'|s,a) \cdot \right. \\ &\qquad \left. \cdot \sum_{a' \in \mathcal{A}} \left[\nabla \pi(s',a') \cdot Q(s',a') + \pi(s',a') \cdot \sum_{s'' \in \mathcal{S}} \gamma \cdot p(s''|s',a') \cdot \nabla V(s'')\right]\right] = \\ &= \sum_{a \in \mathcal{A}} \nabla \pi(s,a) \cdot Q(s,a) + \gamma \sum_{a \in \mathcal{A}} \pi(s,a) \sum_{s' \in \mathcal{S}} p(s'|s,a) \sum_{a' \in \mathcal{A}} \nabla \pi(s',a') \cdot Q(s',a') + \\ &\quad + \gamma^2 \sum_{a \in \mathcal{A}} \pi(s,a) \sum_{s' \in \mathcal{S}} p(s'|s,a) \sum_{a' \in \mathcal{A}} \pi(s',a') \sum_{s'' \in \mathcal{S}} p(s''|s',a') \cdot \nabla V(s'').\end{aligned}$$

De esta manera, se puede observar que el segundo término es igual que el primero pero multiplicado por γ y por la probabilidad de transitar del estado s al estado s' en un paso. Además, el segundo término es de nuevo el primer término, pero multiplicado por la probabilidad de transitar del estado s al estado s'' en dos pasos. Por inducción sobre los estados:

$$\nabla V(s) = \sum_{x \in \mathcal{S}} \sum_{k=0}^{\infty} \gamma^k \cdot \Pr(s \to x, k, \pi) \sum_{a \in \mathcal{A}} \nabla \pi(x, a) \cdot Q(x, a),$$

donde $\Pr(s \to x, k, \pi)$ representa la probabilidad de transitar del estado s al estado x en k pasos, bajo la política π.

Ahora, considerando la ecuación (2.12), así como la distribución de estados de la política dada por la definición 2.1.4, entonces se puede relacionar obtener directamente que:

$$\begin{aligned}
\nabla J(\boldsymbol{\theta}) = \nabla V(s_0) &= \sum_{s \in \mathcal{S}} \left(\sum_{k=0}^{\infty} \gamma^k \cdot \Pr(s_0 \to s, k, \pi) \right) \sum_{a \in \mathcal{A}} \nabla \pi(x, a) \cdot Q(x, a) = \\
&= \left(\sum_{s' \in \mathcal{S}} \sum_{k=0}^{\infty} \gamma^k \cdot \Pr(s_0 \to s, k, \pi) \right) \sum_{s \in \mathcal{S}} \mu(s) \sum_{a \in \mathcal{A}} \nabla \pi(x, a) \cdot Q(x, a) = \\
&\propto \sum_{s \in \mathcal{S}} \mu(s) \sum_{a \in \mathcal{A}} \nabla \pi(x, a) \cdot Q(x, a)
\end{aligned}$$

En particular, para el caso continuo en el que no existe un valor T máximo en el que el proceso se detenga, el término de proporcionalidad es igual a 1, luego se tiene una igualdad:

$$\sum_{s' \to \mathcal{S}} \sum_{k=0}^{\infty} \gamma^k \cdot \Pr(s_0 \to s, k, \pi) = 1 \quad \Longrightarrow \quad \nabla J(\boldsymbol{\theta}) = \sum_{s \in \mathcal{S}} \mu(s) \sum_{a \in \mathcal{A}} \nabla \pi(x, a) \cdot Q(x, a)$$

□

8.2. Apéndice B: Código empleado en las simulaciones

A continuación se proporciona el enlace al repositorio en GitHub que contiene todo el código desarrollado para este trabajo:

- **Repositorio:** https://github.com/samuellozanoiglesias/CoopCoins

Contenido del repositorio

El repositorio incluye el entorno *CoopCoins*, con todo lo necesario para entrenar, analizar y visualizar modelos MARL basados en el entorno *Coin Game*, junto con herramientas de configuración y visualización:

- `coin_game/`: código principal del entorno y motores de entrenamiento.
  - `coin_game.py`: implementación básica del entorno.
  - `coin_game_rllib_env.py`: wrapper para integración con Ray RLlib.
  - `training.py`: scripts de entrenamiento configurables.
  - `analysis.ipynb` y `visualization.ipynb`: notebooks para análisis estadístico y generación de gráficos.
  - `visualize_rllib_models.py`: script para visualizar episodios y generar GIFs.
- `JaxMARL/jaxmarl/`: framework MARL basado en JAX.
- `README.md`: guía completa con instalación, configuración y uso.
- `requirements.txt`: librerías necesarias para reproducibilidad.
- `LICENSE`: licencia MIT.

Instrucciones para reproducir experimentos

Todas las instrucciones están detalladas en el `README.md` del repositorio, pero se resumen aquí los pasos clave:

1. **Clonar repositorio e instalar dependencias:**

```
git clone https://github.com/samuellozanoiglesias/CoopCoins.git
cd CoopCoins
pip install -r requirements.txt
```

2. **Entrenamiento básico:** Usando el script en `training.py`, por ejemplo:

```
from coin_game.training import make_train

config = {
"NUM_ENVS": 4,
"NUM_INNER_STEPS": 300,
"NUM_EPOCHS": 10000,
"LR": 0.001,
"GRID_SIZE": 3,
"REWARD_COEF": [[1.0, 0.0], [1.0, 0.0]],
"PAYOFF_MATRIX": [[1, 0, 0], [1, 0, 0]]
}

params, current_date = make_train(config)
```

3. **Visualización de modelos entrenados:**

```
python coin_game/example_visualization.py --episodes 3
```

Esto permite generar visualizaciones en tiempo real o GIFs que muestran la interacción entre los agentes.

4. **Análisis post-entrenamiento:**
   - Monitoreo de métricas y evolución de recompensas.
   - Comparación de estrategias y actitudes.
   - Generación automática de figuras y estadísticas.

```
jupyter notebook coin_game/analysis.ipynb
jupyter notebook coin_game/visualization.ipynb
```

5. **Configuración adicional:**
   - Modificar `GRID_SIZE`, `NUM_INNER_STEPS`, `NUM_EPOCHS`.
   - Ajustar `REWARD_COEF` para explorar diferentes niveles de cooperación y competitividad.
   - Cambiar la `PAYOFF_MATRIX` para definir otros dilemas sociales.