zy5eteogw

August 14, 2025

1 Temas Tratados en el Trabajo Práctico 1

- Diferencia entre Inteligencia e Inteligencia Artificial.
- Concepto de omnisciencia, aprendizaje y autonomía.
- Definición de Agente y sus características. Clasificación de Agentes según su estructura.
- Identificación y categorización del Entorno de Trabajo en tabla REAS.
- Caracterización del Entorno de Trabajo.

2 Anotaciones

"Acordarse de la definición de agente"

En IA, un agente es cualquier entidad que percibe su entorno mediante sensores y actúa sobre él mediante actuadores, siguiendo algún criterio para maximizar su rendimiento. Puede ser físico (como un robot) o virtual (como un programa).

3 Ejercicios Teóricos

1. Defina con sus propias palabras inteligencia natural, inteligencia artificial y agente.

- Inteligencia Natural: Inteligencia de los seres vivos, los cuales tienen la capacidad de aprender de la experiencia a travez del razonamiento aunque este puede verse afectado por las emociones.
- Inteligencia Artificial: Programa donde se busca que éste tome desiciones y/o aprenda de la experiencia a travez de una retroalimentación.
- Agente: Aquel programa, mecanismo o ambos trabajando en conjunto en donde a travez de una retroalimentación ejecuta o no una acción de forma que se maximize su rendimiento.

2. ¿Qué es un agente racional? Idem anterior

3. ¿Un agente es siempre una computadora? No, puede ser un simple mecanismo el cual ejecute una acción dependiendo del entorno, por ejemplo, un interruptor termomagnetico el cual debido a un estimulo físico (sobrecarga) se ejecuta una acción.

4. Defina Omnisciencia, Aprendizaje y Autonomía.

- Omnisciencia: es el conocimiento completo y absoluto de todas las cosas reales y posibles. Es la capacidad de saber todo lo que ha sucedido, lo que está sucediendo y lo que sucederá.
- Aprendizaje: es el proceso mediante el cual se adquieren, modifican y desarrollan conocimientos, habilidades, conductas y valores a través de la experiencia, el estudio, la instrucción, el razonamiento y la observación.
- Autonomía: es la capacidad de un individuo o entidad para tomar sus propias decisiones, regirse por sus propias normas y actuar de forma independiente, libre de presiones externas o internas.

5. Defina cada tipo de agente en función de su estructura y dé un ejemplo de cada categoría.

- 1. Total o parcialmente observable

Un entorno es totalmente observable si el agente puede percibir el estado completo del mundo a través de sus sensores en cada momento. Un agente en este tipo de entorno no necesita mantener un estado interno para saber lo que sucede.

Un entorno es parcialmente observable si los sensores del agente no pueden acceder a toda la información relevante. La falta de visibilidad puede deberse a sensores defectuosos, ruido o a que simplemente el agente no puede ver ciertas partes del mundo.

- 2. Determinista o estocástico

Un entorno es determinista si el siguiente estado está completamente determinado por el estado actual y la acción del agente. No hay incertidumbre en los resultados.

Un entorno es estocástico si la siguiente acción no se puede predecir con certeza, ya que puede haber factores aleatorios o impredecibles que influyan.

- 3. Episódico o secuencial

Un entorno es episódico si las decisiones del agente en cada "episodio" son independientes de las decisiones en episodios anteriores. Cada acción solo afecta a lo que sucede en el episodio actual.

Un entorno es secuencial si las acciones del agente influyen en los estados futuros y, por tanto, en decisiones posteriores. Es decir, las decisiones tienen consecuencias a largo plazo.

- 4. Estático o dinámico

Un entorno es estático si no cambia mientras el agente está deliberando o pensando en una acción.

Un entorno es dinámico si puede cambiar mientras el agente está en proceso de tomar una decisión.

- 5. Discreto o contínuo

Un entorno es discreto si el conjunto de percepciones y acciones es finito o "enumerado", es decir, se puede contar.

Un entorno es continuo si las percepciones, las acciones o el tiempo son infinitos o tienen un rango de valores.

- 6. Agente indivisual o multiagente

Un entorno es de agente individual si solo hay un agente operando en él.

Un entorno es multiagente si hay dos o más agentes. Estos pueden ser cooperativos (trabajando juntos para un objetivo común) o competitivos (con objetivos opuestos).

6. Para los siguientes entornos de trabajo indique sus propiedades:

a. Una partida de ajedrez

- **Totalmente observable:** Sí, el agente puede ver todas las piezas y sus posiciones en el tablero.
- **Determinista:** Sí, los movimientos de las piezas tienen resultados predecibles y no hay azar.
- **Secuencial:** Sí, cada movimiento afecta las futuras jugadas y el resultado final.
- **Estático:** Sí, porque el entorno solo cambia cuando el agente realiza un movimiento.
- **Discreto:** Sí, el número de movimientos, piezas y estados del tablero es finito.
- **Multiagente:** Sí, hay dos agentes (los jugadores) compitiendo entre sí.

b. Un partido de baloncesto

- **Parcialmente observable:** Sí, un agente (jugador) no puede ver todas las posiciones y acciones de los demás jugadores en todo momento.
- **Estocástico:** Sí, hay un grado de azar en los movimientos de la pelota, los rebotes y las acciones de los otros jugadores.
- **Secuencial:** Sí, las decisiones de un jugador afectan el desarrollo del partido.
- **Dinámico:** Sí, el entorno cambia constantemente debido a los movimientos de los jugadores y la pelota.
- **Continuo:** Sí, las posiciones, velocidades y trayectorias de la pelota y los jugadores son continuas.
- **Multiagente:** Sí, hay varios agentes (los jugadores de ambos equipos) interactuando.

c. El juego Pacman

- **Parcialmente observable:** Sí, Pacman no puede ver lo que hay en los pasillos que no están a su vista.
- **Estocástico:** Sí, el movimiento de los fantasmas puede ser impredecible para el agente (Pacman).
- **Secuencial:** Sí, las acciones pasadas (dónde ha comido los puntos) afectan las futuras decisiones y el resultado del juego.
- **Dinámico:** Sí, el entorno cambia constantemente con el movimiento de Pacman y los fantasmas.
- **Discreto:** Sí, los movimientos son en celdas de una cuadrícula y las acciones son limitadas.
- **Multiagente:** Sí, Pacman interactúa con varios agentes (los fantasmas).

d. El truco

- **Parcialmente observable:** Sí, un jugador no conoce las cartas de los otros jugadores.
- **Estocástico:** Sí, el reparto de cartas es aleatorio, lo que introduce un elemento de azar.
- **Secuencial:** Sí, las decisiones de un jugador (cantar, envidar) tienen un impacto directo en las jugadas posteriores.
- **Dinámico:** Sí, el entorno cambia continuamente con las acciones y el comportamiento de los otros jugadores.
- **Discreto:** Sí, el número de cartas y de jugadas posibles es finito.

- **Multiagente:** Sí, es un juego de múltiples agentes (los jugadores) que pueden formar equipos.

e. Las damas

- **Totalmente observable:** Sí, al igual que el ajedrez, todas las piezas son visibles en el tablero.
- **Determinista:** Sí, cada movimiento tiene un resultado predecible.
- **Secuencial:** Sí, los movimientos de un jugador afectan las futuras posibilidades del otro.
- **Estático:** Sí, porque el tablero no cambia a menos que un jugador haga un movimiento.
- **Discreto:** Sí, el número de movimientos y estados del tablero es finito.
- **Multiagente:** Sí, hay dos agentes (los jugadores) que compiten.

f. El juego tres en raya

- **Totalmente observable:** Sí, el tablero es completamente visible para ambos jugadores.
- **Determinista:** Sí, no hay azar en los movimientos.
- **Secuencial:** Sí, cada jugada influye en las siguientes.
- **Estático:** Sí, ya que el tablero no cambia entre turnos.
- **Discreto:** Sí, el número de movimientos y estados del tablero es finito.
- **Multiagente:** Sí, hay dos agentes (los jugadores) compitiendo.

g. Un jugador de Pokémon Go

- **Parcialmente observable:** Sí, el jugador solo puede ver los Pokémon cercanos y no lo que ocurre en todo el mapa.
- **Estocástico:** Sí, la aparición de Pokémon es aleatoria.
- **Secuencial:** Sí, las decisiones de un jugador (ir a una zona en particular) afectan las futuras recompensas (los Pokémon que podría capturar).
- **Dinámico:** Sí, el entorno real cambia constantemente y los Pokémon aparecen y desaparecen.
- **Continuo:** Sí, la ubicación del jugador y los Pokémon se mueve en un espacio geográfico continuo.
- **Multiagente:** Sí, hay múltiples jugadores interactuando y compitiendo por los mismos recursos.

h. Un robot explorador autónomo de Marte

- **Parcialmente observable:** Sí, el robot no puede ver lo que hay detrás de un obstáculo o más allá de su línea de visión.
- **Estocástico:** Sí, la topografía del terreno es incierta y pueden ocurrir eventos inesperados.
- **Secuencial:** Sí, las decisiones de exploración afectan los futuros descubrimientos y la ruta a seguir.
- **Dinámico:** Sí, el entorno podría cambiar debido a tormentas de polvo o pequeños movimientos de rocas.
- **Continuo:** Sí, los movimientos del robot en el terreno y las mediciones de los sensores son continuos.
- **Agente individual:** Sí, el robot opera por sí mismo para cumplir sus objetivos.

7. Elabore una tabla REAS para los siguientes entornos de trabajo:

a. Crucigrama

Componente	Descripción
Rendimiento	Se mide por la velocidad con la que se completa el crucigrama y la cantidad de respuestas correctas. El objetivo es llenar la mayor cantidad de casillas con las palabras correctas en el menor tiempo posible.
Entorno	Un tablero de crucigrama con palabras cruzadas y pistas. Es un entorno **estático**, **discreto** y **totalmente observable**.
Actuadores	Las acciones del agente, como escribir una letra en una casilla específica o borrar una letra para corregir.
Sensores	La capacidad de leer las pistas, identificar las casillas y percibir las letras que ya se han colocado en el tablero.

b. Taxi circulando

Componente	Descripción
Rendimiento	Se evalúa por la seguridad (evitar accidentes), la eficiencia (llegar al destino en el menor tiempo posible) y la satisfacción del cliente (ofrecer una experiencia de viaje cómoda).
Entorno	Las carreteras, otros vehículos, peatones, señales de tráfico, semáforos, el clima y los pasajeros. Es un entorno **dinámico**, **estocástico** y **parcialmente observable**.
Actuadores	El volante (girar), los pedales (acelerar y frenar), la transmisión (cambiar de marcha) y los indicadores de dirección (señalizar).
Sensores	Cámaras (visión de la carretera), radar (distancia a otros vehículos), GPS (posición), micrófono (interacción con el pasajero) y el motor de temperatura (estado del coche).

c. Robot clasificador de piezas

Componente	Descripción
Rendimiento	Se mide por la precisión (clasificar correctamente las piezas), la velocidad (cantidad de piezas clasificadas por minuto) y la eficiencia energética.

Componente	Descripción
Entorno	Una cinta transportadora con piezas de diferentes formas, tamaños y colores, y los contenedores de clasificación. Es un entorno **episódico** y **estático** (la cinta se mueve, pero el robot no).
Actuadores	Un brazo robótico para recoger las piezas y la capacidad de soltar las piezas en el contenedor correcto.
Sensores	Sensores de visión (cámaras) para identificar la forma, tamaño y color de las piezas, y sensores táctiles para determinar la posición y el agarre de la pieza.

4 Ejercicios Prácticos

8. La Hormiga de Langton es un agente capaz de modificar el estado de la casilla en la que se encuentra para colorearla o bien de blanco o de negro. Al comenzar, la ubicación de la hormiga es una casilla aleatoria y mira hacia una de las cuatro casillas adyacentes. Si…

- … la casilla sobre la que está es blanca, cambia el color del cuadrado, gira noventa grados a la derecha y avanza un cuadrado.
- … la casilla sobre la que está es negra, cambia el color del cuadrado, gira noventa grados a la izquierda y avanza un cuadrado.

Caracterice el agente con su tabla REAS y las propiedades del entorno para después programarlo en Python:

Elemento	Descripción
R (Representación)	La hormiga conoce su orientación (Norte, Sur, Este, Oeste) y el estado de la celda actual (blanca o negra). El plano se representa como una matriz o diccionario donde cada coordenada tiene un color (0 = blanco, 1 = negro).
E (Entorno)	Infinito (o lo suficientemente grande para la simulación), discreto, determinista, **parcialmente observable** (la hormiga solo percibe la celda actual), estático (no cambia por sí mismo, solo por acción del agente).
A (Actuadores)	Cambiar el color de la celda, girar 90° (a la izquierda o a la derecha) y avanzar una celda en la dirección actual.
S (Sensores)	Detectar el color de la celda actual (blanco o negro).

- **Accesibilidad: Parcialmente accesible**, ya que el agente solo puede percibir el estado de la celda en la que se encuentra en ese momento.
- **Determinista: Sí**, las acciones de la hormiga siempre producen el mismo resultado predecible.
- **Episódico: No**, el estado actual del entorno y el comportamiento del agente dependen directamente del historial de acciones y cambios previos.
- **Estático/Dinámico: Estático**, el entorno no cambia por sí solo, solo lo hace como resultado de las acciones de la hormiga.
- **Discreto/Continuo: Discreto**, ya que el entorno está formado por una rejilla finita de celdas individuales.
- **Número de agentes: Un solo agente** (la hormiga).

¿Observa que se repite algún patrón? De ser así, ¿a partir de qué iteración?

Se observa un patron que para la iteracion número 12000 termina la repetición del patrón.

[ ]:
```python
import matplotlib.pyplot as plt
from collections import deque

# Definición de movimientos: N, E, S, O
movs = [(0, 1), (1, 0), (0, -1), (-1, 0)]

# Estado inicial
x, y = 0, 0
direccion = 0  # 0=N, 1=E, 2=S, 3=O
celdas = {}    # Diccionario: (x, y) -> 0 (blanco) o 1 (negro)

# Parámetros de simulación
iteraciones = 20000
limite = 50
velocidad = 0.00001

# Para detección de patrón
historial = deque(maxlen=208)  # 2 ciclos de 104 pasos
inicio_patron = None

# Configuración gráfica
plt.ion()
fig, ax = plt.subplots(figsize=(8, 8))
ax.set_aspect("equal")
ax.set_xlim(-limite, limite)
ax.set_ylim(-limite, limite)
ax.set_title("Hormiga de Langton")

# Lista de puntos negros
negros_x = []
```

```
negros_y = []

# Dibujar los puntos negros iniciales
puntos_negros = ax.scatter(negros_x, negros_y, c="black", s=10)
hormiga_plot, = ax.plot([x], [y], "ro")  # listas, no enteros

for paso in range(iteraciones):
    if not plt.fignum_exists(fig.number):  # si cerraste la ventana
        break

    # Guardar estado para detección de patrón
    color = celdas.get((x, y), 0)
    historial.append((x, y, direccion, color))

    if len(historial) == historial.maxlen:
        mitad = len(historial) // 2
        if list(historial)[:mitad] == list(historial)[mitad:]:
            inicio_patron = paso - mitad
            print(f"Patrón repetitivo detectado a partir de la iteración␣
 ↪{inicio_patron}")
            break

    # Reglas de la hormiga
    if color == 0:  # blanco → gira derecha, pinta negro
        direccion = (direccion + 1) % 4
        celdas[(x, y)] = 1
        negros_x.append(x)
        negros_y.append(y)
    else:  # negro → gira izquierda, pinta blanco
        direccion = (direccion - 1) % 4
        celdas[(x, y)] = 0
        if (x, y) in zip(negros_x, negros_y):
            idx = list(zip(negros_x, negros_y)).index((x, y))
            negros_x.pop(idx)
            negros_y.pop(idx)

    # Avanzar
    dx, dy = movs[direccion]
    x += dx
    y += dy

    # Actualizar puntos negros y hormiga
    puntos_negros.set_offsets(list(zip(negros_x, negros_y)))
    hormiga_plot.set_data([x], [y])

    plt.pause(velocidad)
```

```
plt.ioff()
plt.show()

if inicio_patron:
    print(f"La hormiga entró en el patrón a partir de la iteración␣
 ↪{inicio_patron}")
else:
    print("No se detectó patrón repetitivo en el rango simulado.")
```

9. El Juego de la Vida de Conway consiste en un tablero donde cada casilla representa una célula, de manera que a cada célula le rodean 8 vecinas. Las células tienen dos estados: están *vivas* o *muertas*. En cada iteración, el estado de todas las células se tiene en cuenta para calcular el estado siguiente en simultáneo de acuerdo a las siguientes acciones:

- Nacer: Si una célula muerta tiene exactamente 3 células vecinas vivas, dicha célula pasa a estar viva.
- Morir: Una célula viva puede morir sobrepoblación cuando tiene más de tres vecinos alrededor o por aislamiento si tiene solo un vecino o ninguno.
- Vivir: una célula se mantiene viva si tiene 2 o 3 vecinos a su alrededor.

Caracterice el agente con su tabla REAS y las propiedades del entorno para después programarlo en Python:

¡Claro! Aquí tienes la información sobre el juego de la vida de Conway, formateada en Markdown, tal como lo pediste.

Tabla REAS (Representación, Entorno, Actuadores y Sensores)

Elemento	Descripción
R (Representación)	El sistema mantiene una **matriz (o lista de listas)** que representa el tablero, donde cada celda puede estar en un estado: **0 (muerta)** o **1 (viva)**.
E (Entorno)	Un tablero **bidimensional finito** (o toroidal si los bordes están conectados). Es **discreto** en espacio y tiempo, **determinista**, y **estático** entre cada actualización.
A (Actuadores)	**Actualizar el estado de todas las celdas** simultáneamente basándose en las reglas del juego: nacer, morir o vivir.
S (Sensores)	**Contar el número de células vivas vecinas** (las 8 celdas que rodean a la celda actual).

Propiedades del Entorno

- **Accesibilidad: Totalmente accesible**, ya que el sistema tiene acceso completo al estado de todas las celdas del tablero.
- **Determinista: Sí**, las reglas del juego son fijas y siempre producen el mismo resultado para un estado inicial dado.
- **Episódico: No**, el estado del tablero en un momento dado es completamente dependiente del estado en la iteración anterior.
- **Estático/Dinámico:** El entorno es **estático** entre cada paso de simulación, pero el estado general del sistema es dinámico, ya que cambia con cada actualización.
- **Discreto/Continuo: Discreto**, tanto en el espacio (una cuadrícula de celdas) como en el tiempo (iteraciones o "generaciones").
- **Número de agentes:** Puede interpretarse como un sistema **sin un agente único**. En cambio, funciona como un autómata celular, donde **cada celda actúa como un agente local** que sigue las mismas reglas.

[ ]:
```python
import numpy as np
import matplotlib.pyplot as plt

# Parámetros
filas, columnas = 50, 50
iteraciones = 200
velocidad = 0.1  # segundos entre pasos

# Estado inicial aleatorio
tablero = np.random.choice([0, 1], size=(filas, columnas))

# Configuración de gráfico (se crea solo una vez)
plt.ion()
fig, ax = plt.subplots()
img = ax.imshow(tablero, cmap="binary", interpolation="nearest")
ax.set_title("Juego de la Vida de Conway")

# Función para contar vecinos vivos
def contar_vecinos(tablero, x, y):
    vecinos = [
        tablero[(x-1) % filas, (y-1) % columnas],
        tablero[(x-1) % filas, y],
        tablero[(x-1) % filas, (y+1) % columnas],
        tablero[x, (y-1) % columnas],
        tablero[x, (y+1) % columnas],
        tablero[(x+1) % filas, (y-1) % columnas],
        tablero[(x+1) % filas, y],
        tablero[(x+1) % filas, (y+1) % columnas],
    ]
    return sum(vecinos)

# Simulación
for _ in range(iteraciones):
```

```
    if not plt.fignum_exists(fig.number):  # si cerraste la ventana, cortar␣
 ↪bucle
        break

    nuevo_tablero = np.zeros((filas, columnas), dtype=int)
    for i in range(filas):
        for j in range(columnas):
            vivos = contar_vecinos(tablero, i, j)
            if tablero[i, j] == 1 and vivos in [2, 3]:
                nuevo_tablero[i, j] = 1
            elif tablero[i, j] == 0 and vivos == 3:
                nuevo_tablero[i, j] = 1
    tablero = nuevo_tablero

    img.set_data(tablero)  # actualiza imagen sin redibujar ventana
    plt.pause(velocidad)

plt.ioff()
plt.show()
```

5 Bibliografía

Russell, S. & Norvig, P. (2004) *Inteligencia Artificial: Un Enfoque Moderno.* Pearson Educación S.A. (2a Ed.) Madrid, España

Poole, D. & Mackworth, A. (2023) *Artificial Intelligence: Foundations of Computational Agents.* Cambridge University Press (3a Ed.) Vancouver, Canada