> "En la vida, a menudo las buenas acciones quedan sin recompensa. Sin embargo, con un tamaño de muestra lo suficientemente grande, resultados aparentemente sesgados convergerán a un valor esperado, y las cosas eventualmente saldrán a tu favor."
>
> — *Kaguya-sama: Love is War*, Capítulo 195
> Aka Akasaka

Cadenas de Markov en AI: Una Introducción Sencilla

1. Introducción a las Cadenas de Markov

Las cadenas de Markov son modelos matemáticos que describen una secuencia de eventos donde la probabilidad de cada evento (estado) depende únicamente del estado alcanzado en el evento anterior. Esto se conoce como la **propiedad de Markov**, o **ausencia de memoria**. En términos simples, el estado futuro depende solo del **estado presente** y no de cómo se llegó a ese estado. Formalmente, si (s) y (s') son estados, una cadena de Markov de primer orden satisface:

[
t(s, s') = P(s_{t+1} = s' \mid s_t = s),
]

lo que significa que la probabilidad de pasar al estado *s'* en el siguiente paso de tiempo (t+1) dado el estado actual (s_t) es independiente de los estados pasados antes de (t). Todas estas probabilidades de transición para una cadena de Markov pueden organizarse en una **matriz de transición** (T), donde cada entrada (T_{ij} = P(\text{estado siguiente} = j \mid \text{estado actual} = i)). Cada fila de (T) suma 1 (es una matriz estocástica), reflejando que desde cualquier estado (i), las probabilidades de pasar a todos los posibles estados siguientes (j) suman 100%. Al conjunto de todos los estados posibles se le llama **espacio de estados** (a menudo denotado (S)).

Para adquirir intuición, considera algunos ejemplos cercanos:

PROF

- **Respuestas de WhatsApp:** El hecho de que respondas o no a un mensaje suele depender principalmente del último mensaje que viste (por ejemplo, si es una pregunta o no) y no de todo el historial de chat. Esto puede verse como una cadena de Markov donde el estado es el "tipo de último mensaje" y determina tu respuesta (estado siguiente).
- **IA de videojuegos:** El comportamiento de un NPC podría modelarse con estados como "patrullando", "alerta" o "persiguiendo". La elección de la siguiente acción depende de su estado actual (si está alerta, podría pasar a persecución con cierta probabilidad).
- **Tendencias en redes sociales:** Un tema en Twitter puede estar en estados como "inactivo", "en tendencia" o "viral". Cada día puede mantenerse en tendencia o decaer, dependiendo solo de su estado actual.
- **Lanzamiento de dados y tráfico:** Una tirada de dado justo no tiene memoria (la siguiente tirada es independiente de la anterior), y el flujo de tráfico en un cruce puede modelarse de forma que la congestión del minuto siguiente dependa principalmente del nivel actual (asumiendo que accidentes repentinos o condiciones anteriores más allá del estado actual sean despreciables).

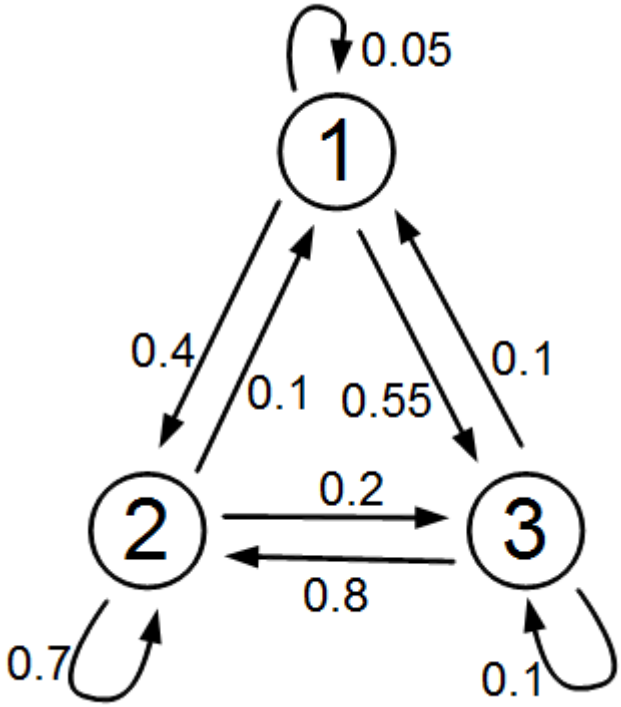


Figura: Ejemplo de una cadena de Markov mostrada como un **diagrama** de transición de estados con tres estados. Las flechas indican transiciones posibles entre estados, etiquetadas con sus probabilidades de transición. Tal diagrama es una representación equivalente a la matriz de transición, ilustrando la propiedad de Markov: desde cada estado (1, 2 o 3), la cadena "salta" al siguiente estado basándose únicamente en las probabilidades de salida del nodo actual, sin recordar dónde estaba antes.

Fuente: Wikimedia Commons - Stochmatgraph

En todos estos casos, el proceso puede pensarse como una secuencia de estados (s_0 \to s_1 \to s_2 \to \cdots) donde cada transición (s_t \to s_{t+1}) sigue una regla de probabilidad fija (P(s_{t+1}\mid s_t)) dada por la matriz (T). Debido a que solo importa el presente, las cadenas de Markov simplifican drásticamente la modelación de fenómenos secuenciales. En lugar de rastrear toda la historia, basta con llevar un registro del **estado actual**. Esto hace que el análisis y la computación sean factibles para sistemas de AI que deben predecir o planificar sobre secuencias, como veremos a continuación.

PROF

2. Diseño de Cadenas de Markov

Al diseñar un modelo de cadena de Markov para un problema de AI, el primer paso es decidir una **representación de estado** clara. Cada estado debe capturar toda la información relevante acerca del pasado que se necesita para determinar el futuro. Si el entorno es **totalmente observable** (el agente tiene acceso al estado verdadero) y **estocástico** (existe aleatoriedad en las transiciones), a menudo podemos modelar su dinámica como una cadena de Markov. Las probabilidades de transición pueden estimarse a partir de datos o definirse mediante conocimiento experto. En esencia, te preguntas: *"Dado el estado X en el tiempo t, ¿cuáles son las probabilidades de pasar a cada estado posible en el tiempo t+1?"* Esas probabilidades forman las entradas de la matriz de transición (T). En un juego de tres en raya, por ejemplo, un "estado" podría ser la configuración del tablero; si aleatorizamos los movimientos de los jugadores, podríamos formar una cadena de Markov de los estados del juego. En un sistema de tráfico, un estado podría ser el nivel de congestión actual (por ejemplo, ligero, medio, intenso) y las transiciones modelan cómo el tráfico tiende a acumularse o dispersarse en la siguiente hora.

Diseñar una cadena de Markov también implica comprender la **estructura del entorno**. En un entorno totalmente observable, el agente sabe exactamente en qué estado se encuentra, de modo que puede usar directamente la cadena de Markov. En un entorno parcialmente observable, el agente podría igualmente **asumir** una cadena de Markov subyacente (esto conduce a los Hidden Markov Models, que veremos más adelante). El entorno puede ser **determinista o estocástico**. Si es determinista (el siguiente estado está fijado dado el estado actual), las probabilidades de transición son 0 o 1; si es estocástico, tenemos una distribución sobre los posibles estados siguientes. Quienes diseñan modelos de AI suelen usar cadenas de Markov para aproximar procesos complejos. Por ejemplo, un juego simplificado por turnos como *Dungeons & Dragons (D&D)* podría tener estados que representan el orden de turno o la fase del juego (exploración, batalla, etc.), y probabilidades que capturen la probabilidad de pasar de una fase a otra a causa de las acciones de los jugadores o tiradas de dados.

En teoría de juegos y entornos multi-agente, las cadenas de Markov pueden modelar la evolución de **estados de juego o estrategias de los jugadores**. Considera un videojuego competitivo con dos jugadores: se puede modelar el estado de la partida (quién va ganando, condiciones especiales activas, etc.) como una cadena de Markov, donde las transiciones ocurren según las estrategias de los jugadores y el azar. Aunque los jugadores elijan acciones, si fijamos una política para cada uno o se emplea una estrategia mixta (probabilística), la *evolución del estado resultante* puede verse como una cadena de Markov. Por ejemplo, en una versión simplificada de piedra-papel-tijera, el estado podría ser la diferencia en el puntaje, y dada la diferencia actual, la partida puede inclinarse a victoria, derrota o empate con ciertas probabilidades (asumiendo ciertos patrones de estrategia). Las cadenas de Markov son útiles en dichos escenarios para analizar el comportamiento a largo plazo o la probabilidad de llegar a ciertos estados (como terminar ganando la partida desde una situación dada).

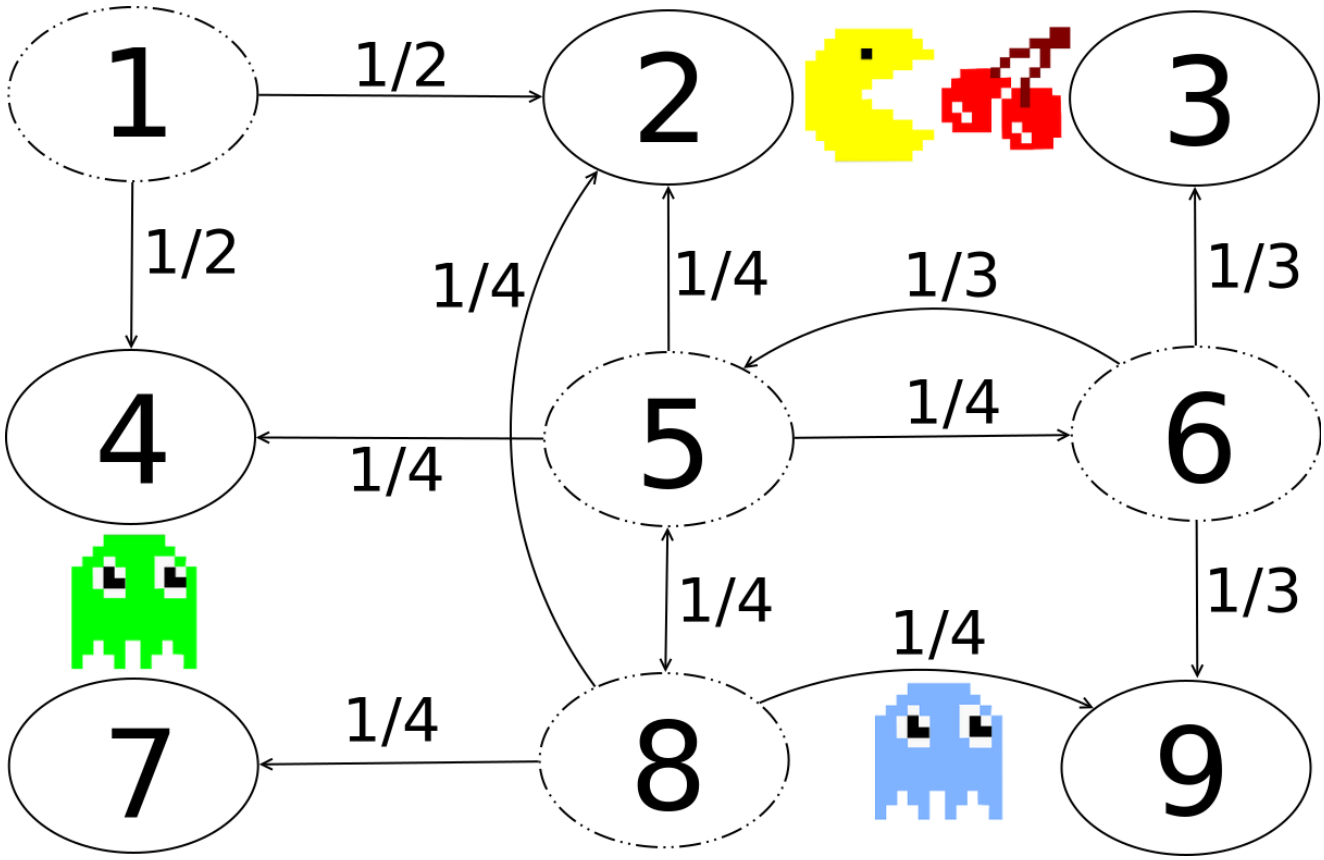


Figura: Diagrama de transición de estados para un videojuego (Pac-Man como ejemplo). Cada óvalo es un estado del juego (regiones numeradas en la cuadrícula de Pac-Man) y cada flecha muestra una transición de estado posible con su probabilidad. Por ejemplo, desde el estado 5 el juego podría pasar al

PROF

estado 6 con probabilidad 1/3 o volver al estado 8 con 1/4, etc., dependiendo del movimiento de Pac-Man y el comportamiento aleatorio de los fantasmas. Estos diagramas ayudan en el **diseño de cadenas de Markov** al visualizar cómo un agente (como Pac-Man) puede desplazarse por el espacio de estados. Al diseñar tu modelo de Markov, te asegurarás de que, desde un estado dado (nodo), las flechas salientes y sus probabilidades reflejen con exactitud las reglas del entorno. Al elegir cuidadosamente los estados y las probabilidades de transición, creamos un modelo de cadena de Markov que captura la dinámica esencial de un sistema o juego.

Fuente: Wikimedia Commons - Transition graph pac-man

3. Cálculo en Cadenas de Markov

Una vez definida una cadena de Markov (espacio de estados y matriz de transición (T)), podemos realizar diversos cálculos para entender su comportamiento. Un concepto central es la **distribución de estado estacionario** (también llamada distribución *stationary* (\pi)). Esta es una distribución de probabilidad sobre los estados que permanece *invariable* bajo el proceso de transición; en otras palabras, (\pi) satisface (\pi T = \pi). Si inicias la cadena en la distribución estacionaria, se mantendrá en esa distribución todo el tiempo. Intuitivamente, a medida que la cadena se ejecuta durante mucho tiempo, "olvida" su estado inicial y la fracción de tiempo que pasa en cada estado converge a las probabilidades de estado estacionario. Por ejemplo, si un estado particular tiene una probabilidad estacionaria de 0.25, a la larga la cadena estará en ese estado alrededor del 25% del tiempo (en promedio). Para encontrar (\pi), se resuelve un sistema de ecuaciones lineales (una por cada estado) o se calcula (T^n) para (n) grande y se observa a qué convergen las filas (o columnas).

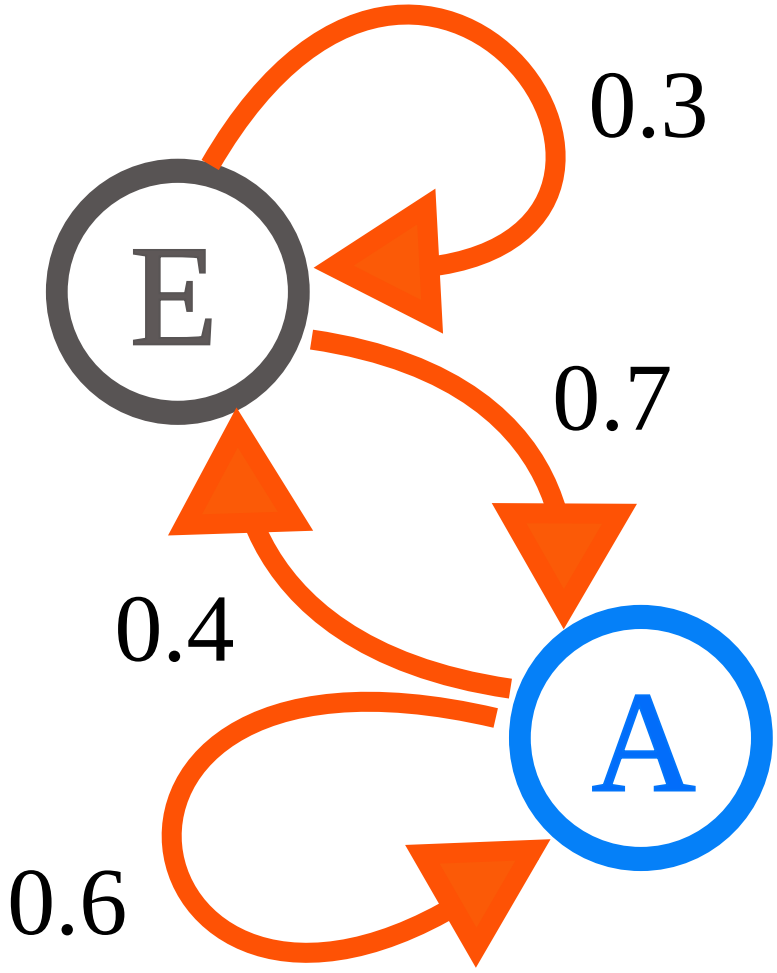
Para calcular las probabilidades de **transición a n pasos** (la probabilidad de ir del estado (i) al estado (j) en exactamente (n) pasos), podemos elevar la matriz de transición a la potencia (n): (T^n). La entrada ((i,j)) de (T^n) da (P(s_n = j \mid s_0 = i)). Esto es útil para preguntas como "Si el sistema está en el estado A ahora, ¿cuál es la probabilidad de que esté en el estado B después de 5 transiciones?" – miras ((T^5)_{AB}). Computacionalmente, se puede hacer multiplicación de matrices repetida o usar algoritmos como **power iteration** (iteración de potencias) que multiplican repetidamente una distribución inicial por (T) (simulando la cadena) hasta converger. La iteración de potencias es una forma sencilla de aproximar la distribución estacionaria: comenzar con cualquier conjetura (\pi^{(0)}) y calcular (\pi^{(k+1)} = \pi^{(k)} T) en cada paso; a medida que (k) crece, (\pi^{(k)}) se acercará a (\pi) si la cadena es *ergódica*.

PROF

La **ergodicidad** es una propiedad que garantiza que una cadena de Markov converja a una única distribución estacionaria. Una cadena ergódica es aquella que es **irreducible** (es posible llegar eventualmente de cualquier estado a cualquier otro estado, al menos en teoría) y **aperiódica** (no queda atrapada en ciclos con un periodo fijo). Si se cumplen estas condiciones, (T^n) a medida que (n \to \infty) se aproxima a una matriz donde cada fila es la distribución estacionaria (\pi). Muchos procesos del mundo real tienen esta tendencia a "olvidar" las condiciones iniciales si se ejecutan lo suficiente y si ningún estado es absorbente. Por ejemplo, considera la navegación web como una cadena de Markov: dada cierta aleatoriedad razonable (la base del algoritmo PageRank de Google), eventualmente la probabilidad de estar en cualquier página web se estabiliza (esa es la distribución estacionaria).

Otros cálculos de interés incluyen los **tiempos de arribo esperados** (por ejemplo, "¿cuántos pasos en promedio tardaré en llegar al estado X empezando en el estado Y?") y las **probabilidades de absorción**

en cadenas que tienen estados absorbentes (estados que, una vez que se ingresa en ellos, no se pueden abandonar). A menudo, esto puede deducirse de (T^n) o resolviendo sistemas de ecuaciones lineales. En resumen, mediante álgebra lineal (potencias de matrices, eigenvalores, eigenvectores) podemos analizar cuantitativamente las cadenas de Markov. Por ejemplo, las probabilidades estacionarias corresponden a un eigenvector de (T) (concretamente, el eigenvector izquierdo con eigenvalor 1). Herramientas como la descomposición en valores propios dan idea del comportamiento a largo plazo de la cadena y de la rapidez con que converge a la estacionariedad (el segundo eigenvalor de mayor magnitud se relaciona con la tasa de convergencia).



PROF

Figura: Una sencilla cadena de Markov de dos estados (Estado A y Estado E). Las flechas indican la probabilidad de pasar de un estado a otro (por ejemplo, la flecha de A a E podría tener probabilidad (p), y de A a A es (1-p)). En una cadena así, es fácil calcular la distribución estacionaria a mano: será la proporción entre las probabilidades de transición opuestas. Por ejemplo, si (P(A \to E) = P(E \to A) = 0.5), la distribución estacionaria será 50% A y 50% E. En cadenas más complejas, usamos algoritmos para encontrar este equilibrio.

Fuente: Wikipedia - Markov chain

4. Inferencia y Agentes de AI

Las cadenas de Markov proporcionan un fundamento para que los agentes de AI *infieran* la dinámica de su entorno y tomen decisiones. Si un agente de AI asume que el mundo cumple la propiedad de Markov, puede simplificar su toma de decisiones centrándose en el estado actual. Diversos tipos de agentes usan el razonamiento con cadenas de Markov de diferentes maneras:

- **Agentes de reflejo simples (Simple Reflex Agents):** Estos agentes eligen acciones basadas únicamente en el estado actual, sin atender al historial. Esto se ajusta perfectamente a la propiedad de Markov. Si la regla del agente dice "si estoy en el estado S, realiza la acción A", en la práctica está aprovechando (aunque sea de forma implícita) la probabilidad de transición (t(S, \cdot)). Por ejemplo, un termostato es un agente de reflejo simple: no le importa si estuvo caliente hace una hora, solo si *ahora* está caliente (estado actual) para decidir encender el aire acondicionado. Podemos modelar la temperatura de la habitación como una cadena de Markov donde la siguiente temperatura depende solamente de la temperatura actual y de las acciones de enfriar/calentar.
- **Agentes de reflejo basados en modelo (Model-Based Reflex Agents):** Estos tienen un **modelo** interno de la dinámica del mundo (un modelo de (t(s,s'))). En otras palabras, el agente construye o recibe una cadena de Markov del entorno. Puede predecir estados siguientes: "Si estoy en el estado S ahora, mi modelo dice que hay un 70% de probabilidad de que pase a S' y 30% a S'' en un paso". Esto usa esencialmente la matriz de transición (T) para la inferencia. Por ejemplo, una IA de videojuego podría conocer las reglas markovianas del juego (cómo suelen comportarse los oponentes desde el estado actual) y actualizar sus creencias de lo que ocurrirá un turno más adelante.
- **Agentes basados en objetivos (Goal-Based Agents):** Estos agentes consideran secuencias futuras de estados para lograr un objetivo. Pueden utilizar las propiedades de las cadenas de Markov para planificar mirando varios pasos hacia adelante (a menudo mediante búsqueda o algoritmos de planificación). Si el agente conoce la matriz de transición (T), puede calcular (T^n) o simular muchas transiciones para ver hacia dónde podría llegar. El razonamiento basado en objetivos implica encontrar una secuencia de transiciones (un camino a través de la cadena de Markov) que conduzca a un **estado objetivo**. En términos de Markov, el agente podría estar interesado en la probabilidad de llegar eventualmente a un estado objetivo (una probabilidad de arribo) o el número esperado de pasos para alcanzarlo desde el estado actual. Al examinar la cadena (a veces usando algoritmos de programación dinámica si hay recompensas de por medio), el agente elige acciones que aumentan la probabilidad de alcanzar el objetivo. En una IA de planificación de rutas para navegación, por ejemplo, el mapa puede verse como una cadena de Markov de ubicaciones, y el agente basado en objetivos busca una ruta de alta probabilidad hacia el destino.
- **Agentes basados en utilidad (Utility-Based Agents):** Estos agentes no solo tienen objetivos, sino también una función de utilidad (valor) para evaluar lo deseable que es cada estado. Cuando se modela un entorno como una cadena de Markov, un agente basado en utilidad combinará las probabilidades de transición con las utilidades de los estados futuros para decidir la mejor acción. Esencialmente, realiza algún tipo de *expectimax* o cálculo de utilidad esperada: considera "si estoy en el estado S y tomo la acción A, el siguiente estado podría ser S' o S'' con ciertas probabilidades (según el modelo de Markov); ¿cuánta utilidad es probable que obtenga?". El agente preferirá la acción que conduzca a la mayor utilidad esperada. En la práctica, esto desemboca en el ámbito de los **Markov Decision Processes (MDPs)**, que son cadenas de Markov extendidas con acciones y recompensas. El agente basado en utilidad, mediante algoritmos como

value iteration o policy iteration, calcula la política óptima sobre la cadena de Markov subyacente de estados.

- **Agentes que aprenden (Learning Agents):** Estos agentes mejoran con el tiempo aprendiendo de la experiencia. En el contexto de cadenas de Markov, un agente que aprende podría comenzar sin conocer las probabilidades de transición y **aprender** estas probabilidades observando la frecuencia de las transiciones de estado (es decir, haciendo una estimación empírica de (t(s,s'))). Esto es exactamente lo que ocurre en **reinforcement learning**, donde un agente explora un entorno y estima el modelo de transición de estados y/o el valor de los estados. Al asumir que el entorno es markoviano, los algoritmos de aprendizaje (como Q-learning o la simulación de Monte Carlo) actualizan las estimaciones basadas en el estado actual y el siguiente estado observado, convergiendo gradualmente a un modelo preciso de la cadena de Markov. Los agentes que aprenden conectan la teoría de juegos y las cadenas de Markov cuando, por ejemplo, ajustan su estrategia en un juego (las probabilidades de transición podrían cambiar a medida que el oponente también aprende – ¡un escenario más complejo!). Aun así, muchos escenarios de aprendizaje en AI se basan en el supuesto de Markov; incluso si el agente no aprende explícitamente "la matriz", a menudo aprende una política que es óptima para la dinámica de la cadena de Markov en el entorno.

Vale la pena señalar que las cadenas de Markov son el peldaño inicial para modelos más avanzados en AI:

- Un **Hidden Markov Model (HMM)** es esencialmente una cadena de Markov donde los estados no se observan directamente (están "ocultos") y solo vemos algunas observaciones emitidas por los estados. La cadena subyacente podría modelar, por ejemplo, el estado emocional de un usuario (feliz, neutral, triste), que no se observa directamente, mientras que las observaciones son sus mensajes o acciones. La AI debe inferir la distribución de estados ocultos usando algoritmos como forward-backward; todo ello se basa en el supuesto de Markov para las transiciones de estado.
- Un **Markov Decision Process (MDP)**, como se mencionó, es una cadena de Markov con acciones y recompensas añadidas. En un MDP, cuando un agente realiza una acción dada en el estado (s), el estado cambia según (P(s_{t+1}=s' \mid s_t=s, a_t=a)). Esto sigue cumpliendo la propiedad de Markov (el siguiente estado depende solo del estado y la acción actuales), y forma la base de cómo modelamos problemas de toma de decisiones para AI. Resolver un MDP (mediante programación dinámica, reinforcement learning, etc.) equivale a encontrar una política óptima asumiendo que las transiciones del entorno son una cadena de Markov influida por las acciones.
- Un **Proceso de Decisión de Markov Parcialmente Observable (POMDP)** generaliza los HMM y los MDP: aquí el agente no observa directamente el estado verdadero, de modo que mantiene una **creencia** (una distribución de probabilidad sobre los estados). Esa creencia en sí puede verse como un estado en una *cadena de Markov aún más grande*. Por lo tanto, las cadenas de Markov también subyacen a la dinámica del estado de creencia: dado el estado de creencia actual (distribución sobre dónde podríamos estar), una acción y una observación, actualizamos a un nuevo estado de creencia (y esa actualización puede verse como una transición en la cadena de estados de creencia).

PROF

En resumen, las cadenas de Markov brindan el marco de razonamiento para que los agentes de AI predigan y planifiquen. Permiten que los agentes **simulen el entorno** en su mente, razonando "si estoy en el estado X ahora, ¿qué es probable que suceda después?" y tomen decisiones informadas. Desde

acciones reflejas hasta planificación estratégica, asumir un mundo markoviano simplifica la complejidad. Quienes investigan y diseñan AI suelen partir de este supuesto porque es manejable matemáticamente y, a menudo, una aproximación razonable a muchos dominios.

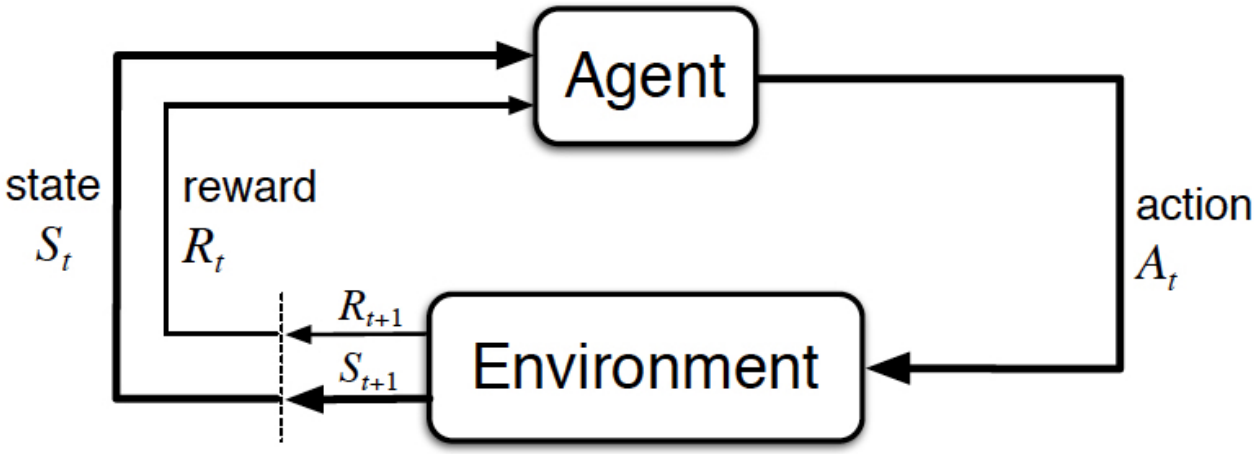


Figura: Ilustración de un agente de AI interactuando con su entorno, que puede modelarse como un proceso de Markov. El agente percibe el **estado** (S_t) y elige una **acción** (A_t); el entorno entonces transita a un nuevo **estado** (S_{t+1}) (siguiendo la dinámica de la cadena de Markov) y proporciona una **recompensa** (R_{t+1}). (En cadenas de Markov puras sin acciones, se ignoran la acción y la recompensa). Este bucle (a menudo representado como en la figura) resalta la propiedad de Markov: la transición del entorno a (S_{t+1}) depende únicamente del estado (S_t) y la acción del agente, sin depender de la historia previa.

Fuente: Oscar Bastardo - Markov Decision Process

5. Conclusión y Próximos Pasos

Las cadenas de Markov son un concepto fundamental en AI y en muchos otros campos porque proporcionan una forma sencilla pero poderosa de modelar procesos secuenciales. En este capítulo introductorio, definimos las cadenas de Markov, junto a temas como los **Hidden Markov Models (HMM)** para datos secuenciales en AI (por ejemplo, habla o texto), los **Markov Decision Processes (MDPs)** para la toma de decisiones óptima en entornos inciertos y los **Procesos de Decisión de Markov Parcialmente Observables (POMDPs)** para enfrentarse a la incertidumbre en la percepción. Todos estos temas avanzados se basan en las ideas aquí presentadas: estados, transiciones y la propiedad de no depender de la memoria pasada más allá del estado actual.

PROF

Para reforzar estos conceptos, consideremos un ejemplo creativo del mundo real en el que una cadena de Markov entra en juego. Imagina los temas en tendencia en una plataforma de redes sociales (como Twitter o TikTok). Podemos definir estados para un tema dado, como: **Inactivo** (casi nadie habla de él), **En tendencia** (está ganando popularidad rápidamente) o **Viral** (extremadamente popular y extendido). Cada día, el tema transiciona entre estos estados con ciertas probabilidades. Tal vez, si un tema está en Tendencia hoy, existe un 50% de probabilidad de que mañana siga en Tendencia, un 40% de que pase a Viral y un 10% de que caiga a Inactivo si el interés decae. Esto puede capturarse en una matriz de transición. Con el tiempo, esto forma una cadena de Markov de la popularidad de un tema. Podríamos usarla para responder preguntas como: "¿Qué fracción de tiempo a largo plazo pasa un tema típico en estado Viral?" o "¿Cuál es la probabilidad de que un meme actualmente Viral esté Inactivo en dos días?"

– todo ello usando los cálculos de los que hablamos (mirando (T^n) o resolviendo la distribución estacionaria). Este ejemplo de dinámica en redes sociales muestra cómo las cadenas de Markov pueden modelar fenómenos cotidianos en AI – en este caso, el comportamiento colectivo de usuarios que impulsa los temas entre distintos estados de popularidad.

A medida que avances, recuerda cómo la **propiedad de Markov** simplifica el aprendizaje y la inferencia. En modelos más complejos (HMMs, MDPs, etc.), a menudo asumimos una cadena de Markov subyacente porque hace que las matemáticas sean manejables y a menudo aproxima la realidad de forma aceptable. En los próximos capítulos profundizaremos en estos temas: por ejemplo, veremos cómo un HMM utiliza una cadena de Markov para modelar secuencias con estados ocultos, y cómo resolver un MDP equivale, en esencia, a encontrar una política óptima en una cadena de Markov de estados cuando hay acciones involucradas. Al dominar las cadenas de Markov, estarás bien preparado para comprender y construir sobre estas técnicas avanzadas de AI.

Las cadenas de Markov demuestran que a veces **la ausencia de memoria es más que suficiente**: al centrarnos en el "ahora", podemos predecir el futuro y tomar decisiones inteligentes sin ahogarnos en datos pasados. Ahora que has sido introducido a las cadenas de Markov, estás listo para explorar el rico panorama de los modelos secuenciales en AI que se basan en este concepto. ¡Feliz aprendizaje al adentrarte en HMMs, MDPs, POMDPs y más allá, armado con el conocimiento de las cadenas de Markov!