Markov Chains

Theory, Applications, and AI Integration

Research Team

Ayman Loay - 320240079 - Sec 3
Introduction & Markov Chain Theory

Youssef Hussein Abdallah - 320240062 - Sec 3
App Navigation Theory

Mohamed Ahmed Mohamed Ali - 320240066 - Sec 3
PageRank and Probability Applications

Ahmed Mahmoud - 320240024 - Sec 1
Markov Decision Processes in AI

Technical Team

Shymaa Mohamed Ahmed - 320240067 - Sec 3

Basmala Refaat Mohamed Azab - 320240022 - Sec 1

Mariam Ahmed Mohamed Ahmed Abdou Ayoub - 320240011 - Sec 1

Aya Yasser Abd El Hamid Hfny - 320240065 - Sec 3

Maryam Ahmed Mohamed Fekry Eltokhey - 320240105 - Sec 4

Front-End Engineer

Ahmed Hossam Aldeen Alsayed Mohammed Hussein - 320240051 - Sec 2

Team Leader

Moaz Hassan Ismail El Garawany - 320240032 - Sec 2

December 2025

1. Introduction to Markov Chains

A Markov chain is a mathematical model for a random process that moves step by step between a set of possible "states," where the probability of the next state depends only on the current state, not on the full history.

Basic Idea

In a Markov chain, time is viewed in discrete steps $n = 0, 1, 2, \ldots$, and at each step the system is in one of a countable set of states (for example: "sunny" or "rainy," or positions on a board). The defining Markov property is that the conditional distribution of the next state given the present and past depends only on the present state, which is often described as "memoryless" with respect to the past.

Formal Definition

A discrete-time Markov chain is a sequence of random variables $X_0, X_1, X_2, \ldots$ taking values in a state space S such that for all times n and states $i_0, \ldots, i_{n+1}$,

$$P(X_{n+1} = i_{n+1} \mid X_n = i_n, \ldots, X_0 = i_0) = P(X_{n+1} = i_{n+1} \mid X_n = i_n)$$

The one-step transition probabilities $P(X_{n+1} = j \mid X_n = i)$ are usually arranged in a transition matrix P, where each row gives a probability distribution over next states from the current state.

Key Components

- **State space:** The set of all possible states, which might be finite or countable (such as weather categories, web pages, or integer positions).
- **Transition matrix:** A matrix $P = (p_{ij})$ where p_{ij} is the probability of moving from state i to state j in one step, with each row summing to 1.
- **Initial distribution:** A probability vector that specifies the starting state (or distribution of starting states) of the chain.

Behavior Over Many Steps

Raising the transition matrix to higher powers (P^n) describes the probabilities of moving between states in n steps. Under suitable conditions (such as irreducibility and aperiodicity), the chain converges to a stationary distribution, a probability vector that remains unchanged by further applications of P.

2. App Navigation Theory

Markov chains model app navigation as user movements between screens, where the next screen depends only on the current one, not past ones. This forgetful trait helps predict user paths in mobile or web apps simply.

Key Ideas in Navigation

States are app screens like home, profile, or settings. The transition table shows chances, such as 0.6 from home to search and 0.4 to profile, with each row adding to 1. Starting chances set the first screen, often login or home.

Tracking User Paths

Raising the transition table to higher powers shows chances over multiple steps, like getting to checkout from home in three taps. Chains that connect fully and avoid cycles settle into a steady pattern of screen visits for better app design.

Real-World Uses

- **PageRank:** Google treats web links as moves to rank pages by long-term visit odds.
- **User Retention:** Spots drop-off spots between screens to improve new user guides.
- **A/B Tests:** Compares move tables between app versions to boost engagement.

Pros and Cons

These models create path heatmaps and smart next-screen hints. Downsides include ignoring strong path links, like in shopping carts.

---

3. PageRank and Probability Applications

PageRank, Google's foundational algorithm, ranks web pages by modeling them as a random surfer following links, with importance derived from incoming link quality and quantity. A small example with four webpages illustrates this: Page A links to B and C, B links only to C, C links to A and D, and D links to A. This setup shows how link structures influence scores.

Link Graph to Matrix

The link graph converts to a transition matrix where each column sums to 1, representing link probabilities from one page. For the four-page example, the matrix entries reflect outgoing link fractions, like Page C splitting its value equally to A and D. This matrix captures the web's hyperlink structure as a Markov chain.

Damping Factor Role

The damping factor, typically 0.85, introduces randomness by letting the surfer jump to any page with probability 1-d (0.15), preventing infinite loops in disconnected graphs. Google's matrix formula becomes $PR(p) = (1-d)/N + d\sum PR(q)/L(q)$, blending link-following with uniform jumps. Lower damping speeds convergence but reduces internal link emphasis.

Convergence Process

PageRank iterates by multiplying the transition matrix with the prior rank vector until changes fall below a tolerance, often converging in 45-52 steps for large networks like Google's early 322 million links. Power iteration ensures stability, with high-rank pages stabilizing slower. Scaling remains nearly linear in log of network size.

SEO Implications

SEO leverages PageRank by earning quality backlinks, as pages with authoritative inbound links gain higher scores, boosting search visibility. Anchor text relevance and link diversity further amplify this, though Google now combines it with hundreds of signals. Internal linking optimizes site flow for better rank distribution.

4. Markov Decision Processes: The Mathematical Core of Sequential AI Control

4.1 Sequential Decision-Making and the MDP Framework

Reinforcement Learning (RL) agents are designed to solve the problem of sequential decision-making—choosing a sequence of actions over time to maximize a long-term accumulated reward. This process must account for outcomes that are partly random (stochastic) and partly controlled by the agent.

The essential mathematical framework governing this interaction is the Markov Decision Process (MDP), which originated in operations research in the 1950s. It transforms the RL challenge into a solvable optimization problem by simplifying the environment's dynamics.

The Markov Property

The MDP relies on the Markov Property, the memoryless principle which states that the future state depends only on the current state and the action taken, and is independent of the entire history of past events. This property is crucial as it reduces the complexity of decision-making from an exponential function of time to a function dependent only on the current state.

4.2 Formalizing the MDP for AI Applications

An MDP is formally defined by five core components, which encapsulate the dynamics of the environment and the agent's control capabilities:

Component	Role in Decision-Making
State Space	The set of all possible environment configurations.
Action Space	The set of commands the agent can execute.
Transition Probabilities	Probability of moving from one state to the next after a specific action.
Reward Function	Immediate scalar feedback for an action/transition.
Discount Factor	Controls how much future rewards are valued (patience).

The primary objective of an RL algorithm is to determine the optimal policy, which is the strategy that selects the best action in every state to maximize the expected long-term return.

4.3 Predictive Models: The Bellman Optimality Equation

The core of finding the best strategy (the optimal policy) lies in solving the Bellman Optimality Equation, which serves as the predictive model for optimal sequential control.

The Optimal Action-Value Function

The key predictive tool is the Optimal Action-Value Function. This function estimates the maximum expected future discounted reward achieved by starting in a particular state, taking an action, and behaving optimally thereafter.

The optimal action is selected by finding the action that maximizes this value function.

The Bellman Optimality Equation

The Optimal Action-Value Function adheres to the Bellman Optimality Equation, a self-consistency relationship that defines the optimal value recursively. This equation is the foundation for algorithms like Q-learning, and it is essential for AI applications because:

- **Optimality:** It ensures the calculation always assumes the best possible subsequent action, thereby guiding the solution toward the ultimate best strategy.
- **Predictive Foundation:** Algorithms like Q-learning use this equation as their update rule, iteratively adjusting the value estimates based on sampled immediate rewards and the predicted maximum future value. This iterative process, known as a temporal difference update, is mathematically guaranteed to converge to the true optimal value, providing the definitive guide for finding the highest possible long-term reward in sequential decision problems.

4.4 Demonstrative Example: Optimal Navigation in a Gridworld

The classic Gridworld problem is a simple, deterministic MDP used to illustrate the principles of optimal policy discovery.

Gridworld Scenario

- **States:** Discrete positions on a 2D grid.
- **Actions:** Deterministic moves: Up, Down, Left, Right.
- **Rewards:** Sparse rewards, such as a large positive value for reaching a goal state and possibly a small negative value for taking any non-terminal step (encouraging efficiency).

Policy Convergence

The solution process (e.g., using Policy Iteration or Value Iteration) demonstrates the predictive power of the Bellman equations. The algorithm starts with an arbitrary strategy and iteratively evaluates and improves it. In the evaluation phase, the high reward from the goal state is diffused backward to neighboring states, assigning higher predicted values to states closer to the goal.

When the algorithm converges, the calculated optimal values stabilize, pointing directly to the optimal path. The resulting optimal policy provides the precise sequence of actions (visualized as arrows on the grid) required to guide the agent perfectly to the terminal state where the maximum cumulative reward is obtained. This simple example confirms that the Bellman Optimality Equation successfully predicts the best long-term course of action in a structured, sequential environment.

References

Section 1 & 2: Introduction and App Navigation

1. "An introduction to Markov chains" – Systematic, textbook-style introduction with clear definitions, examples, and proofs, ideal if you want a rigorous but readable math treatment.

2. Setosa: "Markov Chains explained visually" – Highly intuitive, interactive site that uses animations and diagrams to build the core ideas without heavy formalism, great as a first pass before deeper study.

3. Wikipedia contributors. (n.d.). *Markov chain*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Markov_chain

4. Sekhon, P., & Bloom, D. (n.d.). Applications of Markov Chains. In *Applied Finite Mathematics*. Math LibreTexts. Retrieved from https://math.libretexts.org/Bookshelves/Applied_Mathematics/Applied_Finite_Mathematics_(Sekhon_and_Bloom)/10:_Markov_Chains/10.02:_Applications_of_Markov_Chains

5. Introduction to Markov Chains and Applications. (2016-2017). Chalmers University of Technology. Retrieved from https://www.math.chalmers.se/Stat/Grundutb/CTH/mve220/1617/redingprojects16-17/IntroMarkovChainsandApplications.pdf

6. Introduction to Markov Chain. Shiksha Online Courses. Retrieved from https://www.shiksha.com/online-courses/articles/introduction-to-markov-chain/

Section 3: PageRank Applications

7. Wikipedia contributors. (n.d.). *Markov chain*. Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Markov_chain

8. Sekhon, P., & Bloom, D. (n.d.). Applications of Markov Chains. In *Applied Finite Mathematics*. Math LibreTexts. Retrieved from https://math.libretexts.org/Bookshelves/Applied_Mathematics/Applied_Finite_Mathematics_(Sekhon_and_Bloom)/10:_Markov_Chains/10.02:_Applications_of_Markov_Chains

9. Introduction to Markov Chains and Applications. (2016-2017). Chalmers University of Technology. Retrieved from https://www.math.chalmers.se/Stat/Grundutb/CTH/mve220/1617/redingprojects16-17/IntroMarkovChainsandApplications.pdf

10. Introduction to Markov Chain. Shiksha Online Courses. Retrieved from https://www.shiksha.com/online-courses/articles/introduction-to-markov-chain/

Section 4: AI Applications (Markov Decision Processes)

1. Understanding the Bellman Equation in Reinforcement Learning. DataCamp. Retrieved December 13, 2025, from https://www.datacamp.com/tutorial/bellman-equation-reinforcement-learning

2. Wikipedia contributors. (n.d.). *Markov decision process*. Wikipedia. Retrieved December 13, 2025, from https://en.wikipedia.org/wiki/Markov_decision_process

3. Markov Decision Process (MDP): definition, components and applications. Reinforcement Learning Path. Retrieved December 13, 2025, from https://www.reinforcementlearningpath.com/markov-decision-process-mdp/

4. Markov Decision Process (MDP) in Reinforcement Learning. GeeksforGeeks. Retrieved December 13, 2025, from https://www.geeksforgeeks.org/machine-learning/what-is-markov-decision-process-mdp-and-its-relevance-to-reinforcement-learning/

5. Karpathy, A. (n.d.). REINFORCEjs: Gridworld with Dynamic Programming. Stanford University. Retrieved December 13, 2025, from https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_dp.html

6. Nanade, A. (n.d.). Building a GridWorld Game with Q-Learning from Scratch: Reinforcement Learning. Medium. Retrieved December 13, 2025, from https://medium.com/@nanade.archana/building-a-gridworld-game-with-q-learning-from-scratch-reinforcement-learning-3afc367900a8

7. Policy Iteration. Engineering AI Agents. Retrieved December 13, 2025, from https://pantelis.github.io/aiml-common/lectures/mdp/dynamic-programming-algorithms/policy-iteration/index.html

8. Markov Decision Process in Reinforcement Learning: Everything You Need to Know. Neptune.ai. Retrieved December 13, 2025, from https://neptune.ai/blog/markov-decision-process-in-reinforcement-learning

9. Part 1: Key Concepts in RL. Spinning Up documentation. OpenAI. Retrieved December 13, 2025, from https://spinningup.openai.com/en/latest/spinningup/rl_intro.html

10. Bellman Optimality Equation in Reinforcement Learning. Analytics Vidhya. Retrieved December 13, 2025, from https://www.analyticsvidhya.com/blog/2021/02/understanding-the-bellman-optimality-equation-in-reinforcement-learning/