# Cadenas de Markov con Análisis Computacional

*Nexus-Probability*

**Resumen**

Se explorará el comportamiento de las cadenas de Markov, destacando sus propiedades útiles en diversos contextos. Además, se analizará la complejidad temporal de los algoritmos desarrollados para calcular probabilidades y propiedades en estas cadenas. Se prestará especial atención a la elección de operaciones básicas y a las posibles discrepancias entre la complejidad teórica y la práctica, especialmente al trabajar con probabilidades que pueden convertirse en números extremadamente pequeños.

**Nota 1 (Al Lector:)** *Estas notas están pensadas para ser un complemento y recopilatorio de las técnicas vistas en el curso de procesos estocásticos I, sin embargo, el enfoque será mucho más concentrado en el ámbito computacional y de programación de los algoritmos, asumiendo la mayor parte de la teoría ha sido cubierta en el curso.*

*Aunado a ello, se integrarán algunos ejercicios pensados para programarse, en donde no sólo la teoría si no el modelado de problemas serán de importancia.*

*Sin más preámbulo comencemos por explorar el comportamiento de un tipo de procesos estocásticos que cuentan con una propiedad en extremo útil, las cadenas de Markov.*

**Nota 2 (Acerca de la complejidad de Tiempo.)** *En las secciones por venir iremos analizando la complejidad en tiempo de los algoritmos que vayamos desarrollando para el cálculo de probabilidades y propiedades en cadenas de Markov, por lo que es importante tomar en cuenta las operaciones que se tomarán en cuenta como básicas serán todas las de la aritmética elemental.*

*Como recordatorio, recuerde el análisis de complejidad toma un conjunto de operaciones como básicas para las cuales define su costo en tiempo como constante, $O(1)$. Es decir, nuestro análisis irá entorno a contar la magnitud de la cantidad de éstas operaciones que realiza nuestro algoritmo.*

*Sin embargo, observe suponer ciertas operaciones toman un tiempo constante podría ser engañoso ya que a veces éstas podrían tener un costo mucho más elevado en términos de otras. Por ejemplo, asumiremos la multiplicación de números reales toma $O(1)$, cuando en la práctica sabemos el algoritmo más eficiente de multiplicación tiene complejidad $O(n\log(n))$ donde $n$ es la cantidad de dígitos del número y las operaciones básicas son la suma y la multiplicación de dígitos.*

*Por supuesto, la cantidad de dígitos podría llegar a ser relativamente pequeña durante todo el proceso y por tanto ser este costo despreciable. Pero en lo posterior tenemos que recordar trabajaremos con probabilidades, las cuales pueden rápidamente conventirse en números extremadamente pequeños al preguntarnos por probabilidades en tiempos relativamente cortos. Por lo tanto en lo subsecuente hay que considerar las complejidades corren el riesgo de en la práctica ser más lentas en un factor de $n\log(n)$.*

---

# 1. Indroducción a las Cadenas de Markov

Lo que nos interesa estudiar son cadenas de variables aleatorias $X_0, X_1, \cdots, X_n$, donde la pregunta fundamental es el predecir dada la historia de la cadena, con qué probabilidad podríamos observar un suceso para la variable aleatoria $X_{n+1}$. Sin embargo, esto puede volverse en extremo complicado de hacer ya que $X_{n+1}$ podría tener una fuerte dependencia con las variables previas, haciendo que expresar dicho entrelazamiento se nos haga inmanejable.

Note las cadenas de las cuales hablamos, tal como sugiere la notación $(X_0, X_1, \cdots, X_n)$, son a tiempo discreto, es decir las variables aleatorias deben estar indexadas en el conjunto $T = \{0, 1, 2, \cdots\}$. Y además, para simplificar aún más el comportamiento, se supondrá las variables aleatorias comparten un espacio de estados discreto, $S \subset \mathbb{Z}$.

Es en esta problemática en donde las cadenas de Markov proponen una simplificación entre la dependencia de los eventos del pasado y el evento del presente, considerando cada evento de alguna manera ya encapsula toda la historia que lo llevó a suceder o que simplemente el evento del presente sólo depende de observar lo que sucedió en el instante inmediatamente previo. En ambos casos la propiedad de Markov asegura la probabilidad de observar un evento en el presente dada la historia será la misma que si sólo se condicionara con la información de la variable anterior. En notación:

$$P(X_{n+1} = y | X_0 = x_0, X_1 = x_1, \cdots, X_n = x_n) = P(X_{n+1} = y | X_n = x_n)$$

Donde $x_0, x_1, \cdots, x_n, y$ son realizaciones de sus variables aleatorias respectivamente y pertenecen a $S$.

Ahora bien, note que la fuerza de dicha propiedad es que si estamos en un estado $A$, entonces la probabilidad de transitar en el siguiente paso al estado $B$ será siempre la misma sin importar en que momento nos encontremos.

Esto significa podemos concentrarnos en averiguar todas las probabilidades de transición de un estado a otro $p(x, y)$, $x, y \in S$ y con ellas describir toda la dinámica de la cadena.

Sólo por comparación, note que si nuestro espacio de estados es $\{0, 1, 2\}$ entonces sólo será necesario que encontremos las $3^2$ parejas de posibles transiciones donde una pareja $(x, y)$ representa $P(X_{n+1} = y | X_n = x)$. Mientras que si no se cumpliera la propiedad de Markov entonces en general podríamos tener probabilidades de transición de longitud arbitraria, es decir $(0, 1, 1, 2)$ podría tener una probabilidad distinta de $(1, 2)$ e incluso que $(1, 0, 1, 2)$ a pesar es de la misma longitud y presenta los mismos estados sólo que en distinto orden.

Observe la representación en probabilidad condicional de éstas últimas es:

$$\begin{aligned} &P(X_3 = 2 | X_0 = 0, X_1 = 1, X_2 = 1) \\ &P(X_1 = 2 | X_0 = 1) \\ &P(X_3 = 2 | X_0 = 1, X_1 = 0, X_2 = 1) \end{aligned}$$

respectivamente. Además, note no necesariamente nos será posible escribirlas en general como $P(X_{n+3} = 2 | X_n = 0, X_{n+1} = 1, X_{n+2} = 1)$ ya que esto significaría el estado actual sólo depende de los tres previos pero de haber más historia podría ser dependiera de ésta igual.

Visualmente, la propiedad de Markov tiene que ver con un efecto recursivo o de fractal en el espacio medible. Donde la distribución inicial de los estados para la variable $X_0$ es arbitraria, pero una vez dentro de cualquier estado, la distribución de los estados (no

necesariamente la inicial) se vuelve repetitiva.

De forma gráfica esto lo podemos representar como una partición del espacio en $n$ clases; donde $n$ es el número de estados y a cada uno se le asigna una medida arbitraria. Posterior a ello, cada una de las clases será nuevamente particionada en $n$ sub-clases, donde cada sub-clase tendrá la misma medida cada que parta a una clase proveniente del mismo estado. Y así se seguirá particionando, ahora tomado sub-sub-clases, recursivamente.

Figura 1: Ejemplo de una partición con tres pasos recursivos

En la siguiente imagen simplificamos un poco la partición mostrada arriba y asignamos colores a cada clase para su mejor distinción. El espacio de estados es $T = \{0, 1, 2\}$ y se les asignan a cada uno los colores rojo, azul y verde, respectivamente.

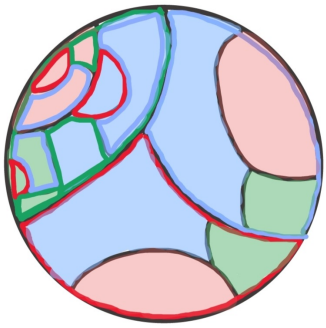

Figura 2: Ejemplo de una partición con tres pasos recursivos

## 2. Representación de una cadena de Markov

Como mencionábamos anteriormente, la propiedad de Markov ayuda enormente a simplificar la dinámica de una cadena, haciendo necesario solamente recordar las $n^2$ posibles transiciones entre los $n$ estados. Los cuales, por conveniencia, guardaremos por ahora en una matriz, aunque hasta el momento dicha matriz no será más que un contenedor

para almacenarlas y accesarlas de forma eficiente.

A esta matriz le llamamos la matriz de transición de la cadena.

$$P = \begin{pmatrix} p(0,0) & p(0,1) & \cdots & p(0,n) \\ p(1,0) & p(1,1) & \cdots & p(1,n) \\ \vdots & \vdots & \cdots & \vdots \\ p(n,0) & p(n,1) & \cdots & p(n,n) \end{pmatrix}$$

Nótese a pesar ya tenemos algo de información que describe a una cadena de Markov, nuestra visualización en el espacio de eventos no es muy conveniente a la hora de querer programarla ni a la hora de querer transitar en ella con fluidez.

Por ejemplo, si quisiéramos describir eventos como lo podría ser $(X_0 = 2, X_1 = 2, X_2 = 0)$ usando nuestra representación actual, la manera más sencilla de modelar la toma de decisiones en cada variable aleatoria, sería a partir de un árbol.

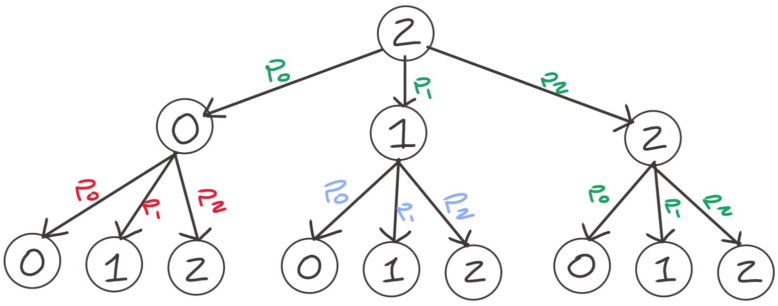


Figura 3: Árbol para la Cadena

En este caso, podemos observar el árbol que se generaría suponiendo $X_0 = 2$. Una vez fija la raíz, los hijos de un nodo en el árbol simbolizan su partición en sub-clases, correspondientes a cada estado, donde cada arista está pesada con su respectiva probabilidad de transición $p(x, y)$. Además, note la profundidad $d$ de cada nivel del árbol simboliza las elecciones para la variable aleatoria $X_d$.

En este modelo de la cadena, podemos ahora ver representado al evento $(X_0 = 2, X_1 = 2, X_2 = 0)$ como el camino en el árbol que satisface las tres condiciones.

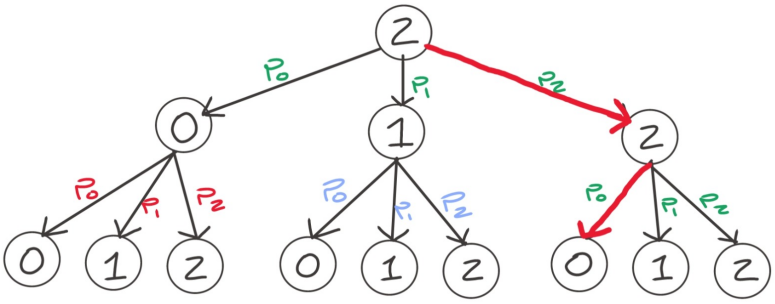


Figura 4: Árbol para la Cadena

Volviendo a la visualización en el espacio de estados, dicho camino puede ser visto como el irse adentrando en sub-clases.

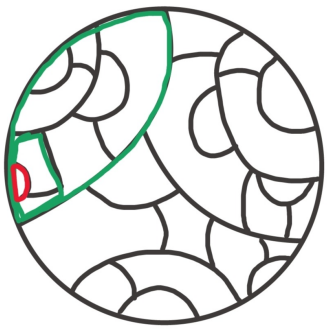

Figura 5: Visualización del Camino en el Espacio de Estados

Aquí hay algo importante a destacar sobre el modelo. Observe que los nodos no están representando el evento de estar en ellos en un cierto tiempo, es decir, el nodo con la etiqueta $0$ al que llegamos en la imagen anterior, no representa al evento $(X_2 = 0)$ el cual se vería representado por los tres caminos que llevan al estado $0$ en la profundidad 2. En este diagrama cada nodo describe un evento conformado por una única elección de estados para cada variable aleatoria en el camino.

En otras palabras, los estados que se desprenden como hijos de distintos nodos están siendo reconocidos como esencialmente estados distintos. En la siguiente imagen podemos apreciar no es lo mismo estar en un estado 0 que en otro, a pesar también es el estado 0.

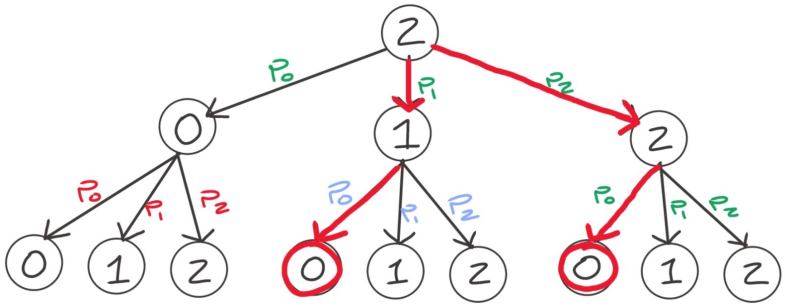


Figura 6: Dos Caminos a Distintos Estados Cero

Todo ello tiene sentido pues los estados 0 en verdad no estaban representando el mismo objeto, son pedazos de la partición diferentes. Sin embargo, nos preguntamos si no habrá una manera de aprovechar la repetición de estados y de probabilidades de transición para compactar el grafo.

Para ello nos aprovechamos fuertemente de la propiedad de Markov para poder condensar todo en un grafo dirigido con solamente $n$ nodos.

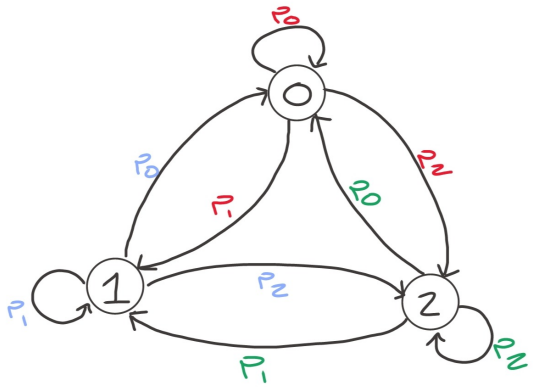


Figura 7: Grafo de una Cadena de Markov

Observe mientras transitamos el grafo vamos precisamente obteniendo la misma perspectiva que cuando alcanzábamos ese mismo estado en nuestro modelo anterior. Moverse en este grafo es igual a moverse en el árbol.

Ahora bien, explicaremos como implementar en Python un grafo dirigido, el cual en particular modelará por lo anterior descrito, una cadena de Markov.

Lo primero que haremos será crear una estructura de lo que será un nodo en nuestro grafo. ¿Para qué hacemos ésto? Bueno, esto lo hacemos porque queremos que cada nodo sea un objeto que guarde información sobre si mismo y que además pueda interactuar y

ejecutar funciones (esto lo explicaremos a más adelante).

Antes de seguir, es importante destacar no es estrictamente necesario trabajar con objetos y podríamos emplear solamente la matriz de transiciones a lo cual se le conoce como representación de un grafo por su matriz de adyacencia, sin embargo, lo que implementaremos será su representación por listas de adyacencia.

En esta representación del grafo lo que se hace es que cada nodo sólo será consciente de aquellos nodos a los que sí puede transitar, en nuestro contexto, aquellos con los que tiene probabilidad de transición distinta de 0. La ventaja de ésta implementación la analizaremos con detalle en un ejemplo más adelante pero la idea es que no transitemos o consideremos caminos innecesarios.

La forma en que construiremos nuestro objeto de nodo será la siguiente:

Primero que nada, a cada nodo le asignaremos un ID o nombre, en nuestro contexto, el nombre o valor del estado que representa.

Después, le asignaremos un atributo correspondiente a su lista de hijos o dicho de otra forma, su lista de ID's de los nodos a los que puede transicionar (nótese sólo es el nombre de éstos y no la probabilidad que tiene de transicionar en ellos). Aunado a ésto, para fácil consulta, le agregaremos un atributo que cuente cuantos hijos tiene.

Finalmente, le añadiremos un método (el cual es una función el objeto será capaz de utilizar) que le permita añadir hijos a su lista y contarlos.

```
1 class state():
2     def __init__(self, name):
3         self.name = name
4         self.children = []
5         self.n_children = 0
6 
7     def add_child(self, child):
8         self.children.append(child)
9         self.n_children += 1
```

Una vez tenemos nuestros objectos que actuarán como nodos, podemos pasar a hacer una función que cree el grafo para nuestra cadena de Markov. Para ello le daremos los siguientes valores:

1. La cantidad de estados
2. La distribución inicial para $X_0$
3. La matriz de transición

La manera en la que recibiremos la distribución inicial será con el siguiente formato, en una línea escribiremos las probabilidades $X_0 = y$ donde $y \in T$ separadas por espacios.

Y para la matriz de transición usaremos el mismo formato para dar cada fila de la matriz.

En cuanto al código, note la variable *states* es una lista que guarda tantos objetos nodo como estados en nuestra cadena, asignándoles su nombre pero dejando sus listas de hijos vacías.

Posteriormente, cuando se recibe la matriz, si el valor en una entrada de la matriz es no negativo se añade el hijo al nodo que corresponde usando el método de nuestro objeto nodo para agregarse hijos.

```
def initialize_chain():
    n_states = int(input("Da la cantidad de estados: "))
    states = [state(i) for i in range(n_states)]

    print("Da la distribucion inicial:")

    initial_dist = list(map(float, input().split()))

    print("Da la matriz de transicion:")

    transition_matrix = []

    for i in range(n_states):
        row = list(map(float, input().split()))
        transition_matrix.append(row)

        for j in range(n_states):
            if row[j] != 0:
                states[i].add_child(j)

    return n_states, states, initial_dist, transition_matrix
```

## 2.1. Análisis de Complejidad

Debido a que recibiremos la matriz de transición, la cual tiene $n^2$ elementos, la complejidad Big O para la función *initialize_chain()* es $O(n^2)$ y esto es algo a tenerse en cuenta para el análisis posterior ya que a pesar tengamos algoritmos quizás lineales para algunas tareas, globalmente no podremos bajar de hacer al menos $n^2$ operaciones.

Aquí cabe destacar nuestra implementación por listas de adjacencia de alguna forma intenta prevenir estemos recurriendo a checar $n^2$ posibles transiciones y nos limitemos

a trabajar únicamente con lo necesario, es decir las aristas no triviales. En general esto lograría que los algoritmos que recorran el grafo estén en términos de $V$ la cantidad de aristas relevantes y nos de una sensación lineal con respecto a la densidad que hay en el grafo, aunque es posible $V = n^2$.

Sin embargo, posteriormente notaremos la utilidad adicional que trae consigo trabajar con la matriz de transición considerándola como tal una matriz y no sólo como un contenedor. Como veremos más adelante, la complejidad de cómputo de probabilidades no está cerca de ser lineal y por ello sacrificar un costo de inicialización de $n^2$ no nos perjudicará.

## 2.2. Ejemplo de una Cadena de Markov

Veamos ahora un ejemplo de una cadena de Markov con tan sólo dos estados y usemos nuestro código para crear su grafo.

Supóngase en una fábrica se tiene una máquina que puede estar ya sea funcionando o descompuesta, cuyo estado se ha observado solamente depende de si funcionaba o no el día anterior. Actualmente la máquina está funcionando y la fábrica ha estimado la probabilidad siga funcionando si estaba funcionando es de 0.3, mientras que la probabilidad funcione tras haber estado descompuesta es de 0.4.

De esta situación nos es fácil encontrar el espacio de estados y la distribución inicial para la variable aleatoria $X_0$. Para el primero decidiremos modelarlo como $S = \{0, 1\}$; donde 0 representa descompuesta y 1 funcionando. Y para obtener la distribución inicial sólo hace falta notar nos dieron por seguro la máquina está en el estado 1 al tiempo 0, es decir $X_0 = (0, 1)$ es la dist. inicial.

Ahora bien, para hallar la matriz de transición nos apoyaremos de un hecho que no hemos destacado hasta ahora y es que la acumulación de las probabilidades de transicionar de un estado fijo a cualquier otro, debe dar 1. Note ésto tiene sentido ya que como dijimos antes, los estados particionan a cada estado. Analíticamente tenemos:

$$\begin{aligned}\sum_{y\in S} p(x,y) &= \sum_{y\in S} P(X_{t+1} = y | X_t = x) \\ &= P(\bigcup_{y\in S}(\{X_{t+1} = y | X_t = x) \\ &= P(\Omega | X_t = x) = 1\end{aligned}$$

De ello podemos inferir la probabilidad de funcionar se descomponga es 0.7 y de estar descompuesta seguirlo estando es 0.6.

$$P = \begin{pmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{pmatrix}$$

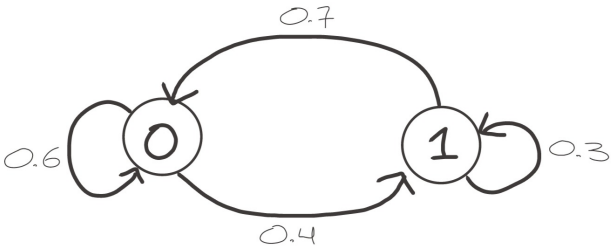


Figura 8: Grafo de la Cadena del Ejemplo de la Máquina

```
n_states, states, initial_dist, transition_matrix =
initialize_chain()
```

```
Da la cantidad de estados: 2
Da la distribucón inicial:
0 1
Da la matriz de transicion:
0.6 0.4
0.7 0.3
```

Para visualizar la estructura del grafo creado, imprimiremos la información de los dos objetos nodos que hay en este caso.

```
def print_information_states(states):
for node in states:
  print("Estado: ", node.name)
  print("Puede transicionar a ", node.n_children, "estados:
", node.children)
return
```

```
print_information_states(states)
```

```
Estado:  0
Puede transicionar a  2 estados:  [0, 1]
Estado:  1
Puede transicionar a  2 estados:  [0, 1]
```

Como otro ejemplo en el que el grafo no sea tan simple, podemos tomar el siguiente, en el cual diremos su dist. inicial es $X_0 = (0, 0, 0, 0, 0.5, 0, 0, 0, 0.5)$:

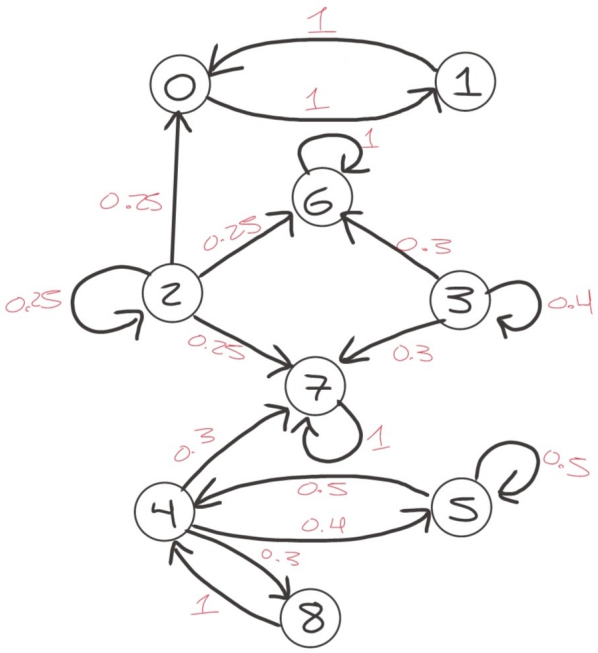


Figura 9: Ejemplo Grafo Cadena de Markov

```
n_states, states, initial_dist, transition_matrix =
initialize_chain()
```

```
Da la cantidad de estados: 9
Da la distribucón inicial:
0 0 0 0 0.5 0 0 0 0.5
Da la matriz de transicion:
0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0
0.25 0 0.25 0 0 0 0.25 0.25 0
0 0 0 0.4 0 0 0.3 0.3 0
0 0 0 0 0 0.4 0 0.3 0.3
0 0 0 0 0.5 0.5 0 0 0
0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0 0
```

```
print_information_states(states)
```

```
Estado:  0
Puede transicionar a  1 estados:  [1]
Estado:  1
```

```
Puede transicionar a  1 estados:  [0]
Estado:  2
Puede transicionar a  4 estados:  [0, 2, 6, 7]
Estado:  3
Puede transicionar a  3 estados:  [3, 6, 7]
Estado:  4
Puede transicionar a  3 estados:  [5, 7, 8]
Estado:  5
Puede transicionar a  2 estados:  [4, 5]
Estado:  6
Puede transicionar a  1 estados:  [6]
Estado:  7
Puede transicionar a  1 estados:  [7]
Estado:  8
Puede transicionar a  1 estados:  [4]
```

# 3. Calculando Probabilidades de Eventos

Ahora que hemos entendido mejor la estructura y modelo para una cadena de Markov, estamos listos para preguntarnos por nuestra tarea principal, la probabilidad de eventos.

Por ejemplo, ¿cuál sería la probabilidad del evento $(X_0 = x_0, X_1 = x_1)$?

Para responder a esa pregunta, será mejor observar un caso particular para el cual supondremos el espacio de estados es $S = \{0, 1, 2\}$. Gráficamente, ¿Cómo observaríamos al evento $(X_0 = 2, X_1 = 0)$?

Recuérdese nuestro modelo ya no es un árbol sino un grafo dirigido en general. Sin embargo, podemos observar las distintas elecciones para las variables aleatorias ordenándolas por profundidad y mostrando las distintas conexiones disponibles de cada nodo. Básicamente estamos elaborando un grafo que nos facilite observar caminos en nuestro grafo de la cadena. A continuación se ilustra este grafo para nuestra cadena con tres estados y hasta el tiempo 3.

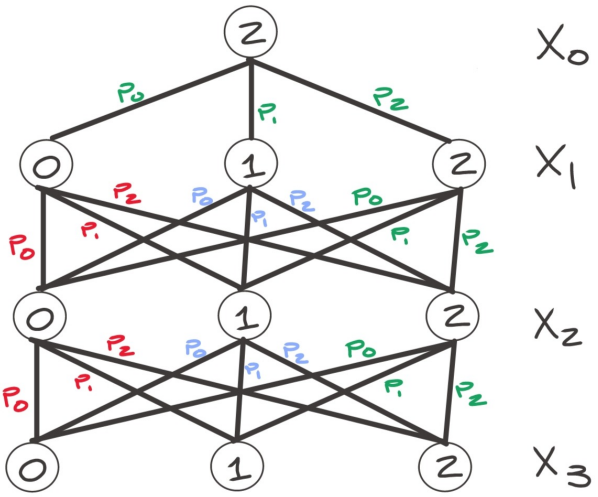


Figura 10: Grafo de los caminos en una cadena de Markov

Es en este grafo que podemos ver representado a nuestro evento.

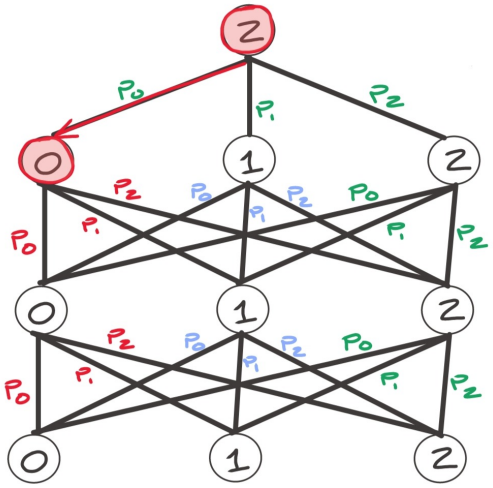

Figura 11: Evento $(X_0 = x_0, X_1 = x_1)$

A través de este diagrama es que pensamos la probabilidad del evento debería ser la probabilidad $X_0 = 2$ multiplicado por la probabilidad de transicionar del estado 2 al 0.

Algebráicamente, denotando por $\Pi_0$ al vector de la distribución de la variable $X_0$, $\Pi_0 = (P(X_0 = 0), P(X_0 = 1), \cdots)$, tenemos:

$$\begin{aligned} P(X_0 = x_0, X_1 = x_1) &= P(X_0 = x_0) * P(X_1 = x_1 | X_0 = x_0) \\ &= \Pi_0(x_0) * p(x_0, x_1) \end{aligned}$$

Para el caso de un evento donde las tres primeras variables aleatorias son fijas $(X_0 = x_0, X_1 = x_1, X_2 = x_2)$ podemos proceder de la misma manera. Tómese por ejemplo el evento particular $(X_0 = 2, X_1 = 0, X_2 = 2)$:

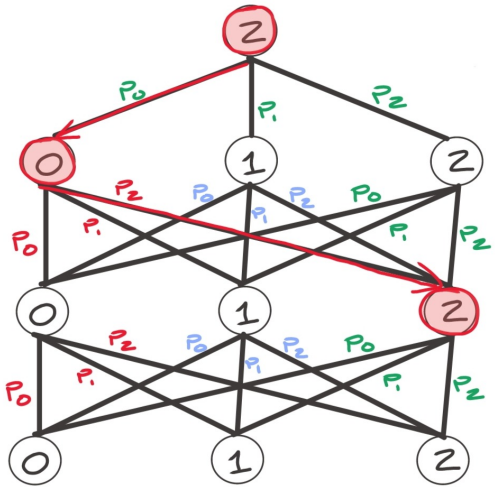


Figura 12: Evento $(X_0 = 2, X_1 = 0, X_2 = 2)$

Algebráicamente:

$$\begin{aligned} P(X_0 = x_0, X_1 = x_1, X_2 = x_2) &= P(X_0 = x_0, X_1 = x_1) * P(X_2 = x_2 | X_0 = x_0, X_1 = x_1) \\ &= P(X_0 = x_0, X_1 = x_1) * P(X_2 = x_2 | X_1 = x_1) \\ &= \Pi_0(x_0) * p(x_0, x_1) * p(x_1, x_2) \end{aligned}$$

Y en general, si tenemos un evento descrito por la elección para las primeras $n$ variables aleatorias, tenemos:

$$P(X_0 = x_0, X_1 = x_1, X_2 = x_2) = \Pi_0(x_0) * p(x_0, x_1) * p(x_1, x_2) * \cdots * p(x_{n-1}, x_n)$$

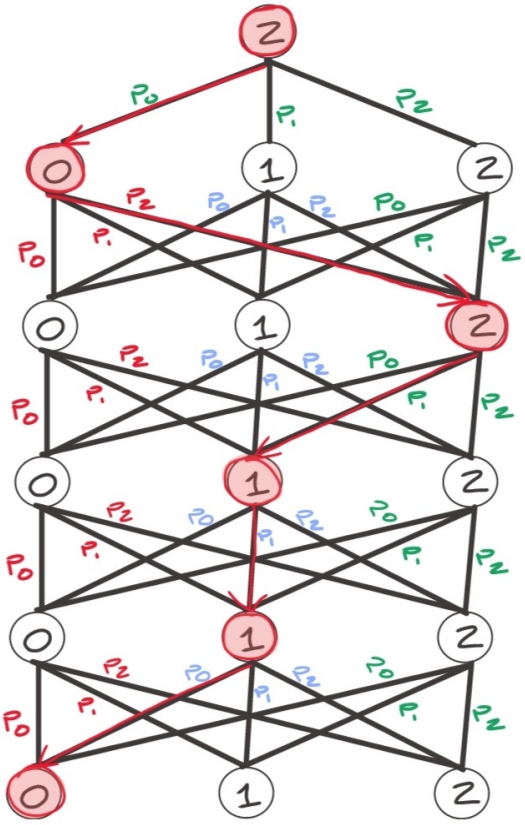

Figura 13: Evento $(X_0 = 2, X_1 = 0, X_2 = 2, X_3 = 1, X_4 = 1, X_5 = 0)$

Pero, ¿qué pasa si no se fija el valor de cada variable aleatoria intermedia? Por ejemplo, ¿cuál es la probabilidad del evento $(X_0 = 2, X_2 = 1)$?

En este caso, parecería se tiene total desinformación sobre el valor de la variable $X_1$, sin embargo lo que el evento $(X_0 = 2, X_2 = 1)$ representa es la unión de todos los eventos, con $X_1$ fijo que cumplen comienzan en el estado 2 y llevan al estado 1.

$$\begin{aligned}(X_0 = x_0, X_2 = x_2) &= (X_0 = x_0, \bigcup_{x_1 \in S} \{X_1 = x_1\}, X_2 = x_2) \\ &= \bigcup_{x_1 \in S} (X_0 = x_0, X_1 = x_1, X_2 = x_2)\end{aligned}$$

Visualmente, esto lo podemos observar como todos los posibles caminos que inician en el estado 2 en la profundidad 0 y llevan al estado 1 en la profundidad 2. Calcular la probabilidad del evento entonces parece se convierte en la suma de las probabilidades de haber elegido cualquiera de los caminos. Note esto tiene sentido, y se verá en la demostración algebraica, ya que esencialemente partimos el evento en un conjunto de eventos disjuntos y contable, lo cual nos permite reconstruir la probabilidad a partir de su suma.

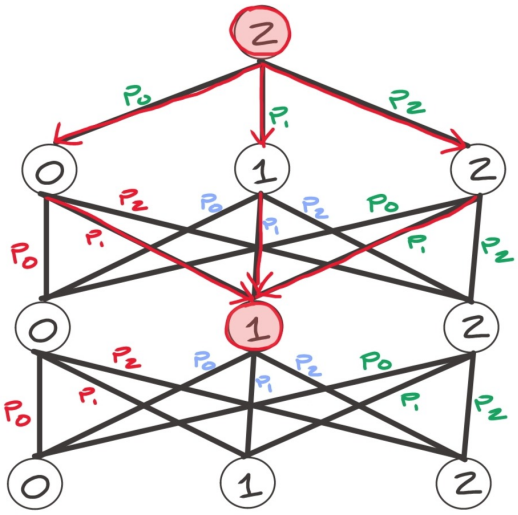


Figura 14: Evento $(X_0 = 2, X_2 = 1)$

---

**Definición 1 (Cadena de Markov)** *Una Cadena de Markov a tiempo discreto es una sucesión de variables aleatorias $X_n,\ n \geq 1$ que toma valores en un conjunto finito o numerable $S$, conocido como espacio de estados, y que satisface la siguiente propiedad*

$$\mathbb{P}(X_{n+1} = j | X_0 = i_0, \ldots, X_{n-1} = i_{n-1}) = \mathbb{P}(X_{n+1} = j | X_n = i_n) \qquad (1)$$

*para todo $n$ y cualesquiera estados $i_0, i_1, \ldots, i_n, j$ en $S$*

La propiedad **(1)** se conoce como la propiedad de Markov.

Recuerde que:

$$\mathbb{P}(X_{t+1} = y | X_t = x) \qquad (2)$$

- La ecuación **(2)** se conoce como la *probabilidad de transición en un paso*. Denotaremos $\mathbb{P}(X_{t+1} = y | X_t = x) = P(x, y)$ con $x, y \in S$
- Para esta clase de procesos suponemos que inicialmente se conoce el comportamiento del sistema, i.e. suponfamos que la distribucion de $X_0$ es conocida y la denotamos por $\Pi : S \to [0, 1]$

  Donde:

  1. $0 \leq \pi(x) \leq 1, \quad \forall x \in S$
  2. $\sum_{x \in S} \pi(x) = 1$

**Nota 3** *Las propiedades de transición en un paso deben cumplir con:*

1. $0 \leq p(x,y) \leq 1, \quad \forall (x,y) \in S$
2. $\sum_{y \in S} p(x,y) = 1, \quad \forall x \in S$ *fijo*

En general, es cómodo designar los estados de la cadena usando los enteros no-negativos $\{0, 1, 2, \dots\}$ y diremos que $X_n$ está en el estado $i$ si $X_n = i$.

La probabilidad de que $X_{n+1}$ esté en el estado $j$ dado que $X_n$ está en el estado $i$ es la *probabilidad de transición* en un paso de $i$ a $j$ y la denotaremos $P_{ij}^{n+1}$:

$$P_{ij}^{n+1} = P(X_{n+1} = j \mid X_n = i)$$

En general, las probabilidades de transición dependen no sólo de los estados sino también del instante $n$ en el cual se efectúa la transición. Cuando estas probabilidades son independientes del tiempo (o sea, de $n$) decimos que la cadena tiene *probabilidades de transición estacionarias o homogéneas en el tiempo*. En este caso $P_{ij}^{n+1} = P_{ij}$ no depende de $n$ y $P_{ij}$ es la probabilidad de que la cadena pase del estado $i$ al estado $j$ en un paso. A continuación sólo consideraremos cadenas con probabilidades de transición estacionarias.

Podemos colocar las probabilidades de transición en una matriz:

$$P = \begin{pmatrix} P_{00} & P_{01} & P_{02} & P_{03} & \cdots \\ P_{10} & P_{11} & P_{12} & P_{13} & \cdots \\ P_{20} & P_{21} & P_{22} & P_{23} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \\ P_{i0} & P_{i1} & P_{i2} & P_{i3} & \cdots \end{pmatrix}$$

que será finita o infinita según el tamaño de $E$. $P$ se conoce como la *matriz de transición* o la *matriz de probabilidades de transición* de la cadena. La $i$-ésima fila de $P$ para $i = 0, 1, \dots$ es la distribución condicional de $X_{n+1}$ dado que $X_n = i$. Si el número de estados es finito, digamos $k$, entonces $P$ es una matriz cuadrada cuya dimensión es $k \times k$. Es inmediato que:

$$P_{ij} = P(X_{n+1} = j \mid X_n = i) \geq 0, \quad \text{para } i, j = 0, 1, 2, \dots$$

$$\sum_{j=0}^{\infty} P_{ij} = \sum_{j=0}^{\infty} P(X_{n+1} = j \mid X_n = i) = 1, \quad \text{para } i = 0, 1, 2, \dots$$

de modo que cada fila de la matriz representa una distribución de probabilidad. Una matriz con esta propiedad se llama una *matriz estocástica de Markov*.