

# Algorítmica

Curso 2023-2024

## Grupo Viterbi



## PRÁCTICA 5- PROGRAMACIÓN DINÁMICA

### Integrantes:

<b>Miguel Ángel De la Vega Rodríguez</b>	miguevrod@correo.ugr.es
<b>Alberto De la Vera Sánchez</b>	joaquinrojo724@correo.ugr.es
<b>Joaquín Avilés De la Fuente</b>	adelaveras01@correo.ugr.es
<b>Manuel Gomez Rubio</b>	e.manuelgmez@go.ugr.es
<b>Pablo Linari Perez</b>	e.pablolinari@go.ugr.es

Facultad de Ciencias UGR  
Escuela Técnica Ingeniería Informática UGR  
Granada  
2023-2024

# Índice general

<b>1 Autores</b>	<b>3</b>
<b>2 Equipo de trabajo</b>	<b>4</b>
<b>3 Introducción</b>	<b>5</b>
<b>4 Descripción del Algoritmo</b>	<b>6</b>
4.1 Modelo Oculto de Markov (HMM) . . . . .	6
4.2 Algoritmo de Viterbi . . . . .	7
4.2.1 Inicialización . . . . .	7
4.2.2 Recursión . . . . .	7
4.2.3 Terminación . . . . .	8
4.2.4 Reconstrucción de la secuencia de estados . . . . .	8
<b>5 Ejemplos de Uso</b>	<b>9</b>
5.1 Lenguaje . . . . .	9
5.1.1 Desambiguación de palabras . . . . .	9
5.1.2 Reconocimiento de voz . . . . .	11
5.2 Predicción de genes . . . . .	12
5.3 Corrección de errores en receptores SDR(Software Defined Radio) . . . . .	14
<b>6 Conclusión</b>	<b>16</b>

## Autores

- **Miguel Ángel De la Vega Rodríguez: 20%**
  - Estructura del documento
  - Ejemplos de Uso
- **Joaquín Avilés De la Fuente: 20%**
  - Explicación del Modelo Oculto de Markov (HMM)
  - Explicación del Algoritmo de Viterbi (incluyendo sus pasos)
  - Documentación de la Introducción
- **Alberto De la Vera Sánchez: 20%**
  - Ejemplo corrección de errores en receptores SDR
  - Conclusión
- **Manuel Gomez Rubio 20%**
  - RELLENAR
- **Pablo Linari Pérez: 20%**
  - Ejemplo de Codificación de Genes

## Apartado 2

### Equipo de trabajo

- **Miguel Ángel De la Vega Rodríguez:** (Ordenador donde se ha realizado el compute)
  - AMD Ryzen 7 2700X 8-Core
  - 16 GB RAM DDR4 3200 MHz
  - NVIDIA GeForce GTX 1660 Ti
  - 1 TB SSD NVMe
  - Debian 12 Bookworm
  - Compilador GCC 12.2.0

## Introducción

El algoritmo de Viterbi es un algoritmo de programación dinámica utilizado para encontrar la secuencia de estados más probable en un **modelo oculto de Markov (HMM, Hidden Markov Model)**. El algoritmo de Viterbi es ampliamente utilizado en diversos campos, como el lenguaje, la ingeniería de comunicaciones, la robótica, la biología, la medicina, la meteorología, etc.

En este documento, presentamos una descripción detallada del algoritmo de Viterbi, su implementación y ejemplos de uso en diferentes campos, haciendo sobre todo énfasis en el uso de la programación dinámica.

## Descripción del Algoritmo

Como bien hemos comentado en la introducción el algoritmo de Viterbi es un algoritmo de programación dinámica utilizado para encontrar la secuencia de estados más probable en un modelo oculto de Markov (HMM, Hidden Markov Model), por lo que en primer lugar pasaremos a explicar dicho modelo así como los elementos que lo componen y su sintaxis matemática, para poder comprender mejor el algoritmo de Viterbi y sus ejemplos.

### 4.1 Modelo Oculto de Markov (HMM)

Un modelo oculto de Markov (HMM) es un modelo estadístico que describe la secuencia de estados a través de la cual pasa un proceso estocástico. En un HMM, el proceso estocástico es un proceso de Markov, lo que significa que la probabilidad de que el sistema pase a un estado futuro depende únicamente del estado actual y no de los estados anteriores, de forma que el cálculo del estado más probable se hace mediante la probabilidad de transición entre estados y de la probabilidad de emisión de observaciones.

Un HMM se representa mediante una tupla  $(Q, V, \pi, A, B, )$  y explicaremos a continuación sus componentes:

- Un conjunto de estados  $Q = \{q_1, q_2, \dots, q_N\}$ .
- Un conjunto de estados observables  $V = \{v_1, v_2, \dots, v_L\}$ .
- Un conjunto de probabilidades de transición entre estados  $A = \{a_{ij}\}$ , donde  $a_{ij} = P(q_t = j | q_{t-1} = i)$ , es decir, la probabilidad de pasar de un estado  $i$  en el instante de tiempo  $t - 1$  a un estado  $j$  en el instante de tiempo  $t$ .
- Un conjunto de probabilidades de las observaciones  $B = \{b_j(v_k)\}$ , donde  $b_j(v_k) = P(o_t = v_k | q_t = j)$ , es decir, la probabilidad de observar  $v_k$  en el estado  $j$  en el instante de tiempo  $t$ .
- Un conjunto de probabilidades de estados iniciales  $\pi = \{\pi_i\}$ , donde  $\pi_i = P(q_1 = i)$ , es decir, la probabilidad de que en el instante 1 (al inicio) se tenga el estado  $i$ .

En dicho modelo, notaremos a la secuencia de observaciones como el conjunto  $O = \{o_1, o_2, \dots, o_L\}$  y al conjunto de estados ocultos con mayor probabilidad como  $S = \{s_1, s_2, \dots, s_L\}$ , donde  $L$  es el número de observaciones, por lo que como se podía deducir se tiene el mismo número de observaciones que estados ocultos en la secuencia con mayor probabilidad, pues estos estados serán consecuencia (en parte) de las observaciones dadas.

## 4.2 Algoritmo de Viterbi

En esta sección procederemos a explicar el **Algoritmo de Viterbi**, donde daremos una explicación detallada de su funcionamiento y del uso de programación dinámica en el mismo.

El algoritmo de Viterbi se basa en la idea de que la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t$  se puede calcular a partir de la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t - 1$ .

El algoritmo de Viterbi se puede resumir en los siguientes pasos:

- **Inicialización:** Calcular la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t = 1$ , es decir, obtener el estado inicial más probable
- **Recursión:** Calcular la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t$  a partir de la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t - 1$ .
- **Terminación:** Calcular la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $T$ .
- **Reconstrucción de la secuencia de estados:** Reconstruir la secuencia de estados más probable a partir de las probabilidades calculadas.

El uso de programación dinámica en este caso es claro, tener una matriz de probabilidades donde se almacenen la probabilidad máxima de cada estado hasta el instante de tiempo  $t$ , lo que nos permitirá calcular la probabilidad máxima de cada estado hasta el instante de tiempo  $t + 1$  de forma eficiente. Por tanto tendremos una matriz  $\mathcal{M}$  de  $N$  filas (una para cada estado  $q_k$  donde  $k \in N$ ) y  $T = L$  columnas (una para cada instante de tiempo  $t \in T = L$ ). Recordemos que teníamos la secuencia de observaciones  $O = \{o_1, o_2, \dots, o_L\}$ .

A continuación procederemos a explicar los pasos del algoritmo de Viterbi de forma más detallada.

### 4.2.1 Inicialización

La matriz  $\mathcal{M}$  se inicializa de la siguiente forma:

- $\mathcal{M}[k][1] = \pi_k \cdot b_k(o_1) \forall k \in N$ , es decir, la probabilidad de que el estado  $k$  sea el estado inicial multiplicado por la probabilidad de que se observe la primera observación en el estado  $k$ . Destacar que  $N$  representa la cantidad de estados posibles ocultos en el modelo oculto de Markov.

### 4.2.2 Recursión

Una vez inicializada la matriz  $\mathcal{M}$ , se procede a calcular las probabilidades de la secuencia de estados más probable hasta el instante de tiempo  $t$  a partir de la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t - 1$ , es decir, para calcular los valores de la columna  $t$  usaremos los de la columna  $t - 1$ , es aquí donde hacemos un uso claro de la programación dinámica. Para ello, se utiliza la siguiente fórmula:

- $\mathcal{M}[j][t] = \max_{i=1}^N \{ \mathcal{M}[i][t-1] \cdot a_{ij} \cdot b_j(o_t) \}$ , es decir, se multiplica la probabilidad máxima del estado  $i$  en el instante  $t - 1$  por la probabilidad de transición de  $i$  a  $j$  por la probabilidad de observar  $o_t$  en el estado  $j$ , todo esto para todo  $i \in N$  y se selecciona el máximo de estos valores. Tenemos así la máxima probabilidad de llegar al estado  $j \in N$  en el instante de tiempo  $t$ .

### 4.2.3 Terminación

En esta última fase el objetivo será obtener cual es la máxima probabilidad de llegar a un estado  $k \in N$  en el instante de tiempo  $t = T$ , es decir, la probabilidad de la secuencia de estados más probable hasta el instante de tiempo  $t = T$ . Para ello, se utiliza la siguiente fórmula:

- $P = \max_{i=1}^N \{\mathcal{M}[i][T]\}$ , es decir, se selecciona el máximo de las probabilidades de llegar a cada estado en el instante de tiempo  $t = T$ .

donde tenemos que  $P$  almacena la probabilidad máxima de todos los posibles estados  $k \in N$  en el instante de tiempo  $t = T$ .

### 4.2.4 Reconstrucción de la secuencia de estados

Una vez obtenida la probabilidad máxima de la secuencia de estados más probable hasta el instante de tiempo  $t = T$ , se procede a reconstruir la secuencia de estados más probable. La idea a desarrollar es la siguiente:

- En el instante  $t = T$  se selecciona el estado  $k$  que maximiza la probabilidad de llegar a dicho estado en el instante de tiempo  $t = T$ , es decir, se selecciona el estado  $k$  tal que  $\mathcal{M}[k][T] = P$ , ya obtenido dicho valor obtendremos el estado  $s$  en el instante  $t - 1$  que maximiza la probabilidad de llegar a  $k$  en el instante de tiempo  $t = T$ , es decir, se selecciona el estado  $s$  tal que  $\mathcal{M}[s][T - 1] \cdot a_{sk} \cdot b_k(o_T) = \mathcal{M}[k][T] = P$ , y así sucesivamente hasta llegar al instante de tiempo  $t = 1$ .

Tenemos ahora la idea con la que obtendremos dicha secuencia más probable de estados, es decir, la secuencia de estados  $S = \{s_1, s_2, \dots, s_L\}$  que maximiza la probabilidad de la secuencia de observaciones  $O = \{o_1, o_2, \dots, o_L\}$ . Veamos por tanto como vamos a hacerlo:

- En el instante de tiempo  $t - 1$  se selecciona el estado  $i \in N$  y se multiplica dicho valor por la probabilidad de transición de  $i$  a  $k$  y por la probabilidad de observar  $o_t$  en el estado  $k$ , si dicho valor nuevo obtenido coincide con el valor de la matriz  $\mathcal{M}[k][T] = P$ , entonces se selecciona el estado  $i$  como el estado anterior a  $k$  en la secuencia de estados más probable. En caso contrario se hace dicha comprobación para todos los estados  $i \in N$ , obteniendo así el estado  $s$  en el instante  $t - 1$  de la secuencia de estados ocultos con mayor probabilidad.
- Este proceso se repite para todo  $t \in T$  hasta llegar al instante de tiempo  $t = 1$ , obteniendo así la secuencia de estados ocultos con mayor probabilidad.



## Ejemplos de Uso

El algoritmo de Viterbi es ampliamente utilizado en diversos campos tales como el lenguaje, la ingeniería de comunicaciones, la robótica, la biología, la medicina, la meteorología, etc. A continuación presentamos algunas de las aplicaciones específicas de este algoritmo en distintos campos.

### 5.1 Lenguaje

El algoritmo de Viterbi es utilizado en el reconocimiento de voz, en el reconocimiento de escritura a mano, en la corrección de errores en el texto, en la traducción automática, en la generación de texto, en la síntesis de voz, en la transcripción de audio, etc.

#### 5.1.1 Desambiguación de palabras

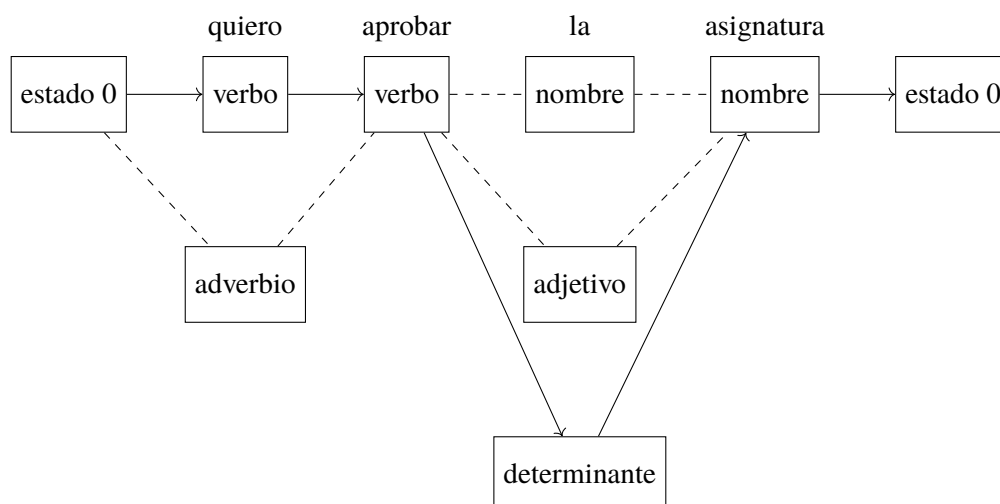
Cuando queremos procesar el lenguaje natural, es común encontrarnos con palabras que tienen múltiples significados. Por ejemplo, la palabra *privado* está reconocida por la RAE como un adjetivo, un sustantivo y un verbo. Para desambiguar estas palabras cuando procesamos un texto, podemos utilizar el algoritmo de Viterbi,

#### Definición del modelo oculto de Markov (HMM)

En este caso particular, los elementos del modelo oculto de Markov serían:

- El conjunto  $Q$  de estados ocultos (categorías gramaticales)
- El conjunto  $V$  de estados observables (palabras)
- El conjunto  $A$  de probabilidades de transición entre estados (probabilidades de cambio de categoría gramatical, por ejemplo, que un nombre vaya detras de un verbo)
- El conjunto  $B$  de probabilidades de emisión de observaciones (probabilidades de que una palabra pertenezca a una categoría gramatical, por ejemplo, que la palabra *perro* sea un sustantivo es mucho mayor a que sea un adjetivo)

Mostramos un ejemplo para el texto *quiero aprobar la asignatura*:



Donde, los observables son la secuencia de palabras *quiero aprobar la asignatura* y los estados ocultos son las categorías gramaticales de las palabras, que como se puede ver, contemplan sólo un conjunto limitado de categorías gramaticales. Esto se debe a que la probabilidad de pertenencia de determinadas palabras a ciertas categorías gramaticales es 0, lo que simplifica y acelera el proceso de desambiguación.

### Aplicación del Algoritmo de Viterbi

El algoritmo de Viterbi se utiliza para encontrar la secuencia más probable de estados ocultos (categorías gramaticales) dada una secuencia de observaciones (palabras). Para nuestro ejemplo, el proceso se desarrolla de la siguiente manera:

- Inicialización:
  - Definir los estados ocultos posibles. Por ejemplo, en nuestro caso, los estados posibles son verbo, nombre, determinante, etc.
  - Definir las probabilidades iniciales para cada estado oculto. Por ejemplo, es muy probable que la primera palabra *quiero* sea un verbo.
  - $\pi(\text{verb}) = P(\text{verb}|\text{estado}_0)$ .
- Recursión:
  - Para cada palabra en la secuencia, calcular la probabilidad de que cada posible estado oculto (categoría gramatical) siga a cada estado anterior. Esto se hace utilizando las probabilidades de transición ( $A$ ) y las probabilidades de emisión ( $B$ ).
  - Ejemplo: Para la palabra *aprobar*, se calcularía  $P(\text{verb}_2|\text{verb}_1) \cdot P(\text{aprobar}|\text{verb}_2)$  para todas las categorías posibles de *aprobar*.
  - Se selecciona el estado que maximiza esta probabilidad.
- Terminación:
  - Una vez procesadas todas las palabras, se selecciona la secuencia de estados que maximiza la probabilidad total de la secuencia observada.
  - Esta secuencia representará las categorías gramaticales más probables para la oración.
- Reconstrucción de la secuencia de estados:

- Utilizando las probabilidades calculadas, se reconstruye la secuencia de categorías gramaticales más probable.
- Por ejemplo: quiero (verbo), aprobar (verbo), la (determinante), asignatura (nombre).

### 5.1.2 Reconocimiento de voz

Dada una secuencia de observaciones acústicas, queremos predecir la secuencia de las palabras que se obtendrán.

#### Definición del modelo oculto de Markov (HMM)

- Estados Ocultos (Q): Fonemas (sonidos básicos del habla)
- Observaciones (V): Características acústicas extraídas de la señal de voz (vectores de características como Mel-Frequency Cepstral Coefficients - MFCC)
- Probabilidades de Transición (A): Probabilidades de transición entre fonemas.
- Probabilidades de Emisión (B): Probabilidades de emitir una característica acústica específica dado un fonema.

#### Aplicación del algoritmo

Mel-Frequency Cepstral Coefficients - MFCC estudia los diferentes tonos, acentuaciones, etc. que tienen las palabras dentro de los idiomas, lo que permite a las IAs aprender con mayor precisión los lenguajes mediante señales de audio

- Inicialización:  
Definir Estados Ocultos posibles. En este caso, son los diferentes fonemas que se pueden emplear en los idiomas  
  
Definir las probabilidades de cada estado oculto.  $\pi$  (fonema): Probabilidad inicial de cada fonema. Estas probabilidades pueden estar basadas en la frecuencia de los fonemas al inicio de palabras en el idioma.
- Recursión:  
Seleccionado un elemento cualquiera, sin que sea el primero, de la secuencia, hay calcular la probabilidad para cada estado oculto posible. Para cada segmento de la señal de voz (cada observación acústica), calculamos la probabilidad de que cada posible estado oculto (fonema) siga a cada estado anterior utilizando las probabilidades de transición y de emisión.  
  
y se seleccionará el estado que maximice la probabilidad.
- Terminación:  
Una vez procesados los fonemas que son empleados en las palabras, se selecciona la secuencia de estados que maximiza la probabilidad total de la secuencia obtenida. Obteniendo así, la secuencia de fonemas con mayor probabilidad
- Reconstrucción de la secuencia de estados:  
Se usan las probabilidades calculadas para reconstruir la secuencia de fonemas más probables.

Usamos las probabilidades calculadas para reconstruir la secuencia de fonemas más probable, y luego mapeamos estos fonemas a palabras usando un diccionario fonético. Ejemplo Práctico

Imaginemos que la secuencia de observaciones acústicas es  $O=o_1, o_2, o_3$  y los fonemas posibles son  $/k/$ ,  $/ae/$ ,  $/t/$ , que corresponden a las palabras  $\text{çat}$ ". Inicialización Recursión

Y así sucesivamente para  $/ae/$  y  $/t/$ . Terminación Obtendremos la secuencia con mayor probabilidad Reconstrucción Se reconstruye esta secuencia obteniendo que la secuencia de fonemas más probable es  $/k/ \rightarrow /ae/ \rightarrow /t/$ , que mapeamos a la palabra  $\text{çat}$ ".

Este ejemplo muestra cómo el algoritmo de Viterbi se utiliza para convertir una secuencia de observaciones acústicas en una secuencia de fonemas, y luego en palabras, en el reconocimiento de voz.

## 5.2 Predicción de genes

En esta sección, presentamos un ejemplo de uso del algoritmo de Viterbi en la predicción de genes en secuencias de ADN. La predicción de genes es un problema importante en bioinformática, En general, la predicción de genes trata de localizar en las largas secuencias de ADN, y de forma automatizada, las subsecuencias de nucleótidos que conforman los diferentes genes.

### Descripción del problema

Existen cuatro tipos de nucleótidos, que se suelen representar por las letras A, C, G y T en función de la base nitrogenada que contengan: Adenina, Citosina, Guanina o Timina. Una cadena compuesta por estas cuatro letras representa la estructura primaria de una molécula de ADN. Por tanto el algoritmo se utiliza para identificar exones que poseen la información necesaria para la síntesis de proteínas. Los modelos ocultos de Markov (HMM) son una herramienta común en la predicción de genes debido a su capacidad para manejar secuencias donde los estados exones e intrones (sirven de mensajeros para el código de los exones) son ocultos y solo las secuencias de ADN observables están disponibles.

### Definición del modelo oculto de Markov (HMM)

Primero definimos los elementos del modelo oculto de Markov:

- El conjunto  $Q$  de estados ocultos (exones e intrones)
- El conjunto  $V$  de estados observables (nucleótidos)
- El conjunto  $A$  de probabilidades de transición entre estados (probabilidades de cambio de exón a intrón y viceversa)
- El conjunto  $B$  de probabilidades de emisión de observaciones (probabilidades de observar una  $A$  en un exón)

El modelo de Markov oculto dado, trata de capturar las diferencias estadísticas en exones e intrones. El modelo tiene cuatro estados, donde  $E_1$ ,  $E_2$ , y  $E_3$  se utilizan para modelar las propiedades estadísticas básicas en los exones. Cada estado  $E_i$  utiliza un conjunto de probabilidades de emisión diferentes para reflejar el símbolo en la posición  $i$  de un codón (secuencia de 3 nucleótidos ej: ATC). El estado  $I$  se utiliza para modelizar los intrones.



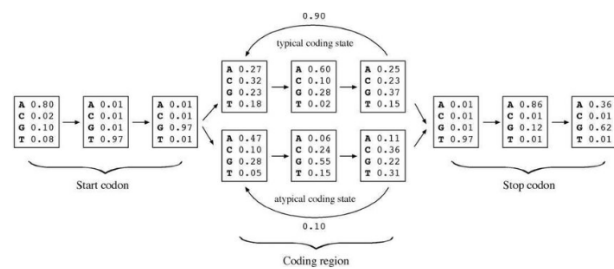


Figura 5.2: Cálculo para una procariota

## Integración en el software

Existen varias herramientas bioinformáticas que implementan el algoritmo de Viterbi para la predicción de genes como GeneMark que utiliza Cadenas Ocultas de Markov para identificar genes en células procariotas y eucariotas. Glimmer es otra herramienta que utiliza el algoritmo de Viterbi para la predicción de genes en genomas bacterianos. Estos programas suelen integrar datos adicionales y heurísticas para mejorar la precisión de la predicción, adaptándose a las características específicas del genoma en estudio. En resumen, el algoritmo de Viterbi es una técnica poderosa para la predicción de genes, aprovechando su capacidad para manejar secuencias y estados ocultos en un contexto biológico complejo ya que muchas veces se necesita modelizar el comportamiento futuro del ADN para detectar genes y regiones codificantes y así poder detectar enfermedades genéticas y tratarlas a tiempo.

## 5.3 Corrección de errores en receptores SDR(Software Defined Radio)

Es una tecnología para comunicaciones inalámbricas que se enfoca en la especificación en software de los elementos de un sistema de comunicación. Una de las características de una SDR, es la detección y corrección de errores, que nos permite un uso eficiente del ancho de banda asignado, pudiendo así aumentar el número de usuarios que el sistema puede soportar.

### Descripción del problema

En general, en sistemas de comunicación inalámbrica, la corrección de errores es esencial para asegurar la integridad de los datos transmitidos. La radio definida por software(SDR) permite diseñar terminales móviles reconfigurables que operan bajo múltiples estándares de comunicación, como GSM(Global System Mobile) y DCS(Digital System Communication). Estos sistemas utilizan la codificación convencional para la detección y corrección de errores, y el algoritmo de Viterbi se utiliza para la decodificación eficiente de estos códigos

### Elementos del modelo oculto de Markov (HMM)

Primero definimos los elementos del modelo oculto de Markov:

- El conjunto  $Q$  de estados ocultos (las posibles combinaciones de bits en el codificador)
- El conjunto  $V$  de estados observables (datos codificados recibidos)
- El conjunto  $A$  de probabilidades de transición entre estados (probabilidades de transición entre estados, generados por la combinación de los distintos bits)

- El conjunto  $B$  de probabilidades de emisión de observaciones (probabilidades de que una observación (bit recibido) sea emitida desde un estado particular, es decir, la probabilidad de recibir una secuencia de bits dados los estados del codificador)

### Aplicación del algoritmo

El algoritmo de Viterbi se utiliza para encontrar la secuencia más probable de estados ocultos (bits originales) dada una secuencia de observaciones (bits recibidos). A continuación, se describe el proceso detallado:

- Inicialización:
  - Definir los estados ocultos: Los estados son las posibles combinaciones de bits que el codificador puede generar .
  - Por ejemplo, en un codificador con  $N = 4$  bits, hay  $2^N = 16$  estados posibles.
  - Definir las probabilidades iniciales: Asignamos una probabilidad inicial a cada estado oculto.
  - Por ejemplo, es probable que el estado inicial sea 0000.
- Recursión:
  - Cálculo métrico de la rama (Branch Metrics): Para cada bit recibido, calculamos la distancia Hamming (métrica utilizada para comparar dos secuencias de símbolos del mismo tamaño, determinando cuántas posiciones en las secuencias difieren entre sí) entre los bits recibidos y las posibles secuencias de bits generadas por las transiciones de estado.
  - Por ejemplo, para una secuencia recibida "11", calculamos la distancia Hamming con todas las transiciones posibles
  - Selección de transiciones sobrevivientes: Determinamos las transiciones con la menor métrica acumulada en cada paso.
  - Por ejemplo, tomando la secuencia anterior, calculamos  $P(00|11) \cdot P(11|\text{estado})$  para todas las categorías posibles.
  - Actualizar métricas: Acumulamos las métricas de distancia para cada posible trayectoria (aquí es donde toma relevancia la programación dinámica).
- Terminación:
  - Una vez procesados todos los bits recibidos, seleccionamos la secuencia de estados (bits originales) que maximiza la probabilidad total
  - Esta secuencia representará los bits originales más probables
- Reconstrucción de la secuencia de estados:
  - Utilizamos las probabilidades calculadas para reconstruir la secuencia de bits originales
  - Por ejemplo, si la secuencia más probable de estados es 10 01 00 11 , esta es la secuencia de bits original decodificada

Una vez visto la aplicación del algoritmo más detallada, podemos decir que el algoritmo de Viterbi para la corrección de errores se basa en: dada una secuencia de bits recibida (la cual puede contener ruido, además de estar codificada), el algoritmo de Viterbi calcula las métricas de rama (distancia Hamming) y selecciona las transiciones con menor métrica acumulada. Finalmente, reconstruye la secuencia de bits originales más probable

## Conclusión

Esta práctica nos proporciona un acercamiento al algoritmo Viterbi, sobre todo en el ámbito de la programación dinámica visto a través de diferentes ejemplos. Podemos destacar de dicho algoritmo que puede resolver eficientemente problemas complejos de secuenciación en diversos ámbitos. De este modo, conseguimos que el lector de dicha memoria, en este caso para alumnos, no solo comprenda el funcionamiento del algoritmo, sino que también vea de una forma más cercana sus aplicaciones en problemas reales.

Este análisis teórico y desarrollo de ejemplos prácticos refuerzan la importancia de la programación dinámica como herramienta esencial en la resolución de problemas de optimización y secuenciación, mostrando como puede ser de gran utilidad en diversos campos de la ciencia e ingeniería.