

Clase nº 14
Año académico 2003

Parte I: Diseño de sistemas de ASR basados en HMM

Rita Singh

Facultad de Informática
Carnegie Mellon University

Índice de contenidos

- ◆ Reconocimiento de palabras aisladas
 - Clasificación bayesiana
 - Decodificación basada en las mejores secuencias de estado

- ◆ Reconocimiento de secuencias de palabras
 - Clasificación bayesiana
 - Decodificación basada en las mejores secuencias de estado
 - Mejor decodificación de ruta y gráficos plegados

- ◆ Estrategias de búsqueda
 - Primero en profundidad
 - Primero en amplitud

- ◆ Estructuras óptimas de grafos
 - Decodificación del lenguaje restringido
 - Decodificación del lenguaje natural
 - HMM para el lenguaje
 - Unigrama
 - Bigrama
 - Trigramas

Clasificación de patrones estadísticos

- ◆ Dados los datos X , hallar a cuál de un número de clases pertenecen, basados en distribuciones de datos conocidos desde

$$C_1, C_2, \dots, C_N \\ \cup C_2, \text{ etc.}$$

- ◆ Clasificación bayesiana:

$$\text{Clase} = C_i : i = \operatorname{argmax}_j P(C_j)P(X|C_j)$$

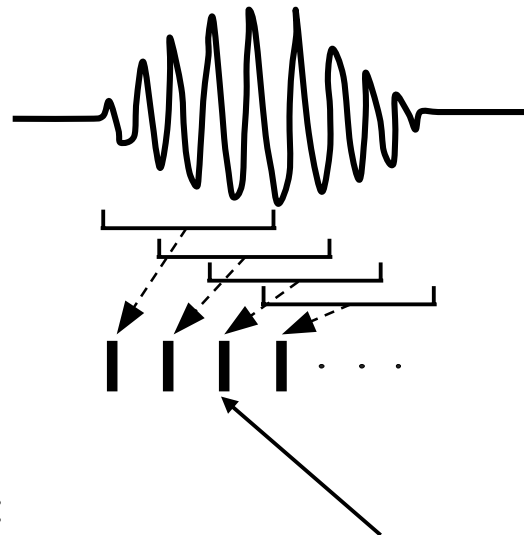
probabilidad *a priori* de C_j

Probabilidad de X dada por la distribución de probabilidad de C_j

- ◆ La probabilidad *a priori* da cuenta de las proporciones relativas de las clases
 - Si nunca vió ningún dato, acertaría la clase basándose en estas probabilidades por si solas
- ◆ $P(X|C_j)$ explica las pruebas obtenidas a partir de los datos observados X

Clasificación estadística de palabras aisladas

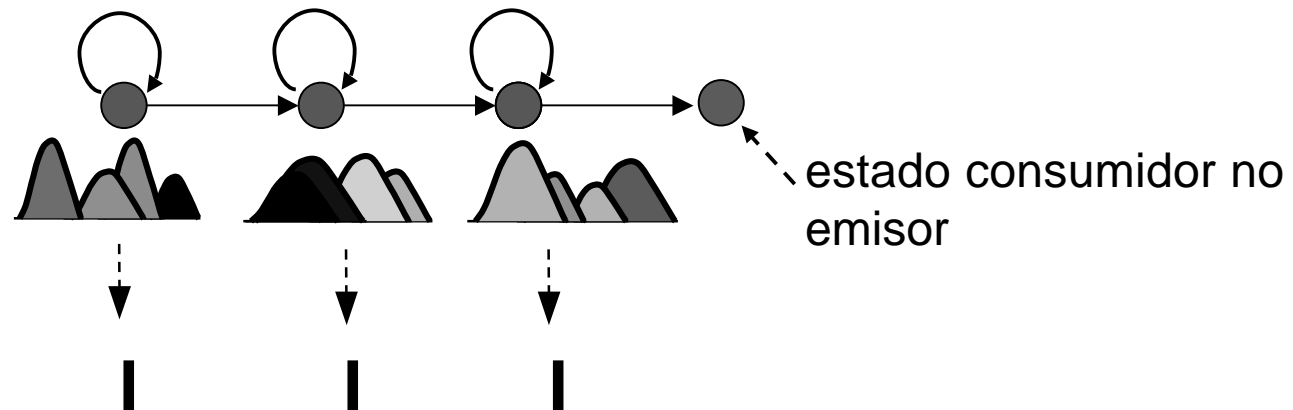
- ◆ Las clases son palabras
- ◆ Los datos son ejemplos de palabras habladas aisladas
 - La secuencia de vectores característicos derivados de la señal de voz, típicamente 1 vector desde un tramo de voz de 25ms, con tramos cambiados por 10ms.



- ◆ Clasificación bayesiana:
$$\text{Palabra_Reconocida} = \underset{\text{word}}{\text{argmax}} P(\text{word})P(X|\text{word})$$
- ◆ $P(\text{word})$ es una probabilidad *a priori* de *palabra*
 - Obtenida a partir de nuestra expectativa de la frecuencia relativa de aparición de la palabra
- ◆ $P(X|\text{word})$ es la probabilidad de X computada sobre la función de distribución de word

Computando $P(X|word)$

- ◆ Para computar $P(X|word)$, debe existir una distribución estadística para X correspondiente a $word$
 - Cada palabra debe estar representada por algún modelo estadístico
- ◆ Representamos cada palabra mediante un HMM (modelo oculto de Markov)
 - Un HMM es realmente una forma gráfica de función de densidad de probabilidad para los datos con variación en el tiempo

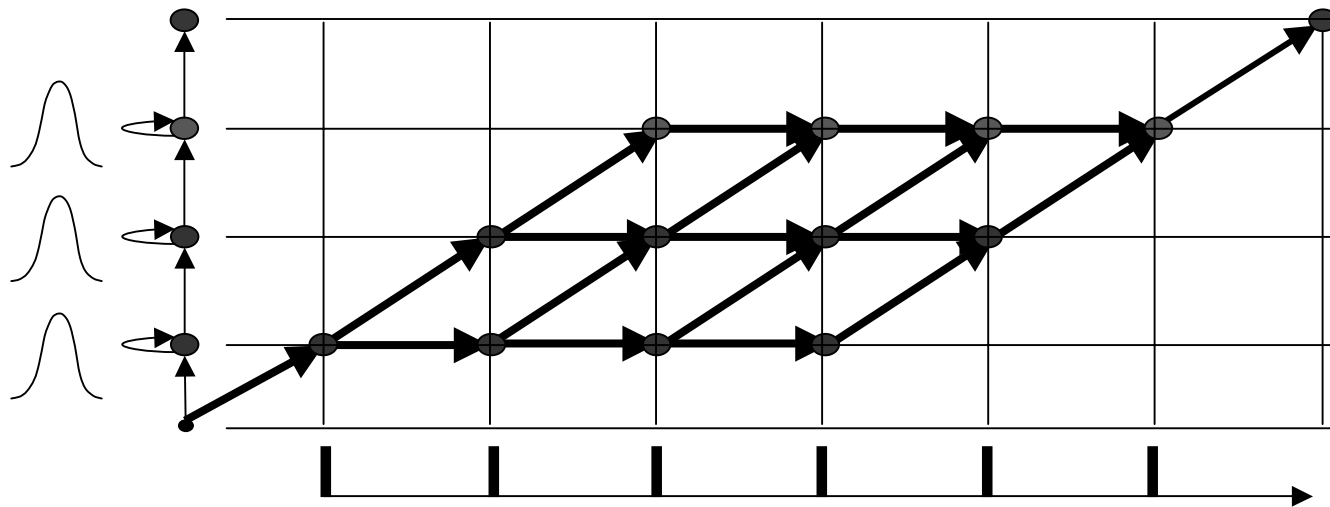


- ❑ Cada estado posee una función de distribución de probabilidad
- ❑ Las transiciones entre estados están gobernadas por probabilidades de transición
- ❑ En cada instante de tiempo, el modelo se encuentra en algún estado, y emite un vector de observación desde la distribución asociada con dicho estado

Computando $P(X|word)$

- ◆ La secuencia de estado real que generó X nunca se conoció
 - $P(X|word)$ debe por tanto tener en cuenta *todas* las secuencias de estado posibles

$$P(X | word) = \sum_{s \in \{\text{todas las secuencias de estado}\}} P(X, s | word)$$



Las probabilidades de todas las secuencias de estado posibles deben añadirse para obtener la probabilidad total

Computando $P(X|word)$

- ◆ La secuencia real de estado que generó X nunca se conoció
 - $P(X|word)$ debe por tanto tener en cuenta *todas* las secuencias de estado posibles

$$P(X | word) = \sum_{s \in \{\text{todas las secuencias de estado}\}} P(X, s | word)$$

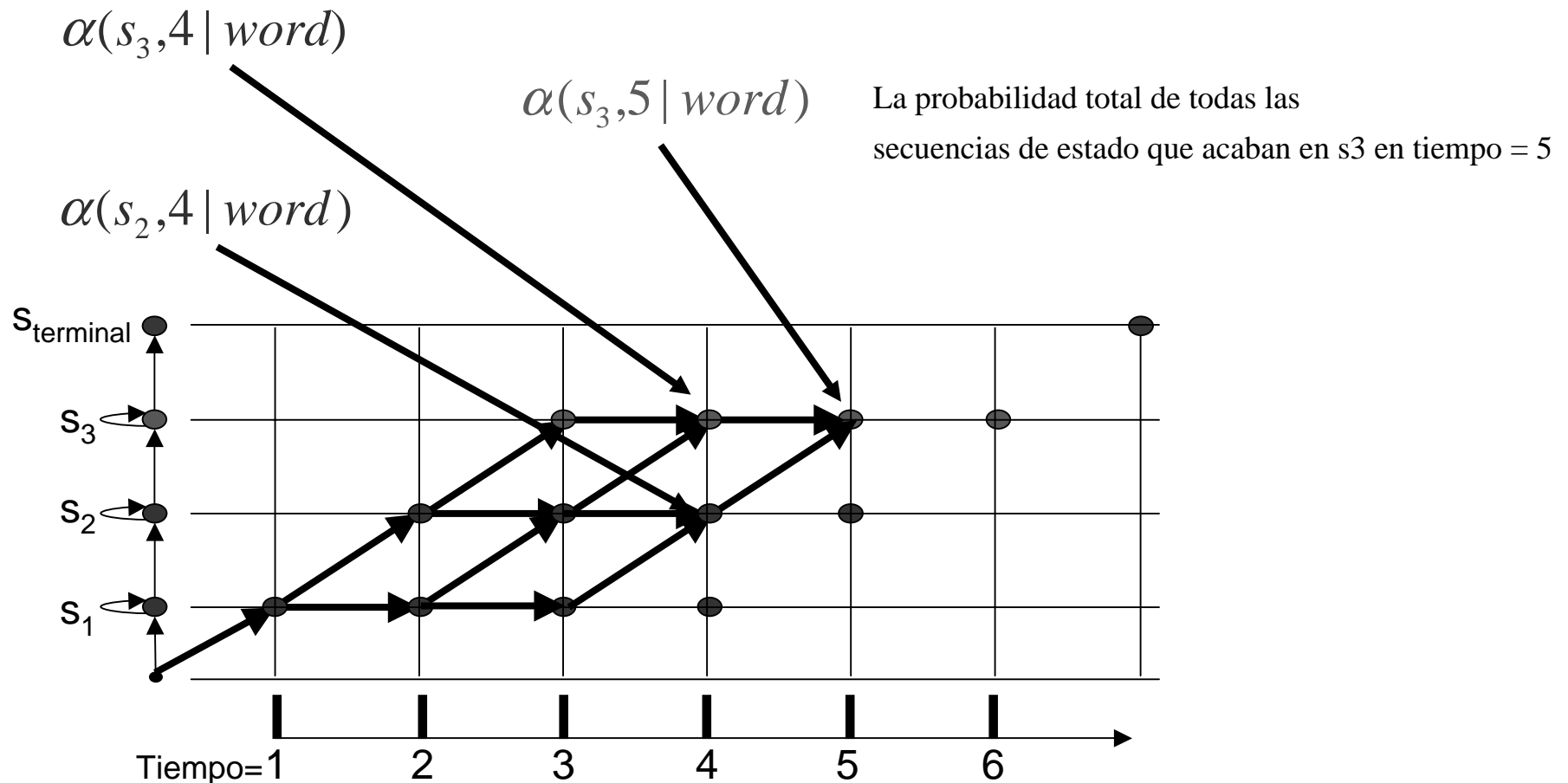
- ◆ El número real de secuencias de estado puede ser muy largo
 - No puede sumar explícitamente todas las secuencias de estado
- ◆ $P(X|word)$ puede sin embargo calcularse eficientemente mediante la recursión hacia delante

$$\alpha(s, t | word) = \sum_{s'} \alpha(s', t-1 | word) P(s | s') P(X_t | s)$$

La probabilidad total de todas las secuencias de estados que acaban en el estado s en tiempo t

Computando $P(X|word)$

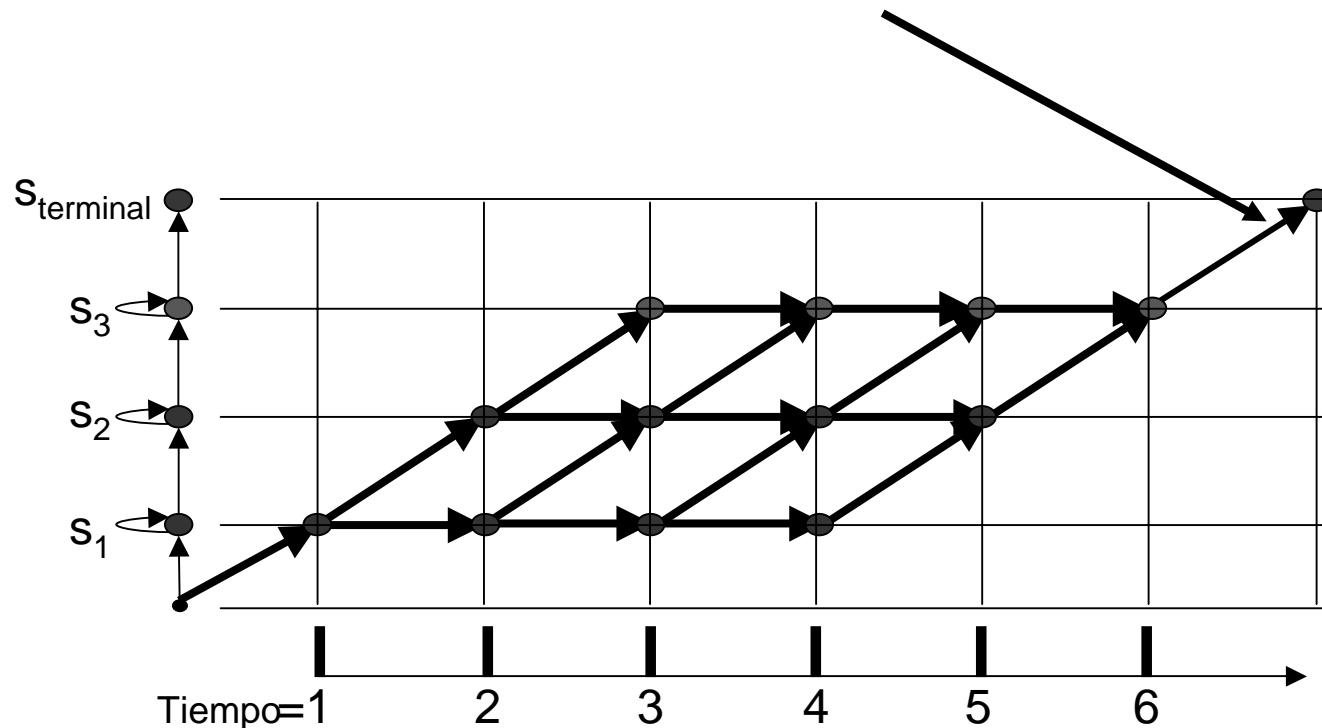
◆ La recursión hacia adelante



Computando $P(X|word)$

- ◆ La probabilidad total de X , incluyendo las contribuciones de todas las secuencias de estado, es la probabilidad de avance al final del nodo no emite.

$$P(X | word) = \alpha(s_{terminal}, T | word)$$



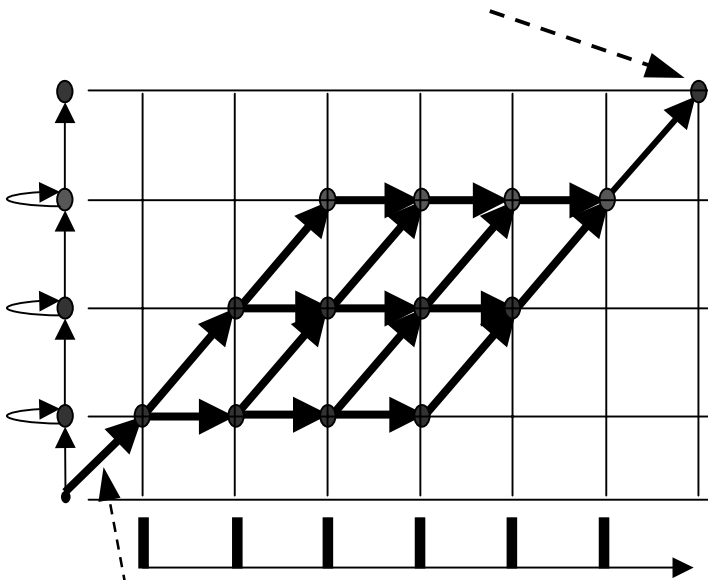
Decodificación de palabras aisladas

Clasificación entre dos palabras: *Odd* y *Even*

HMM para *Odd*

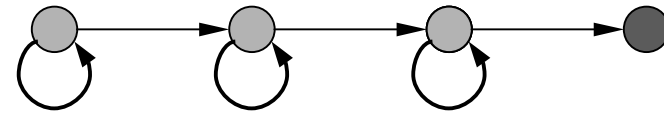


$$P(\text{Odd})P(X|\text{Odd})$$

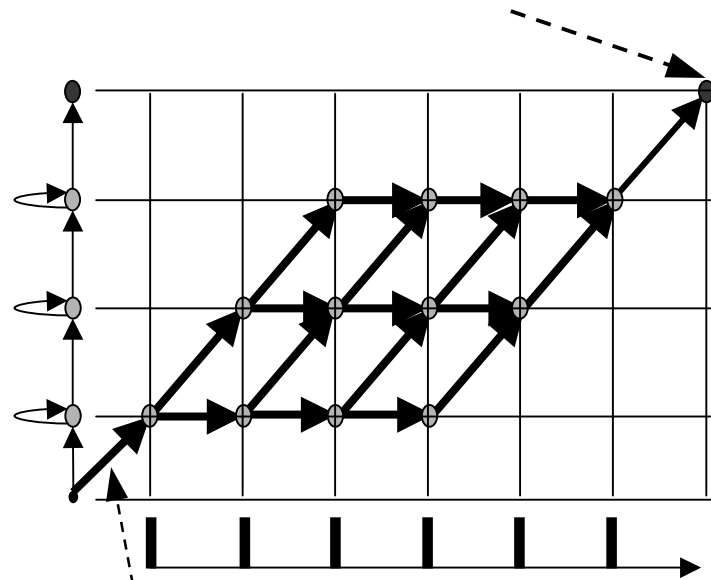


$$P(\text{Odd})$$

HMM para *Even*

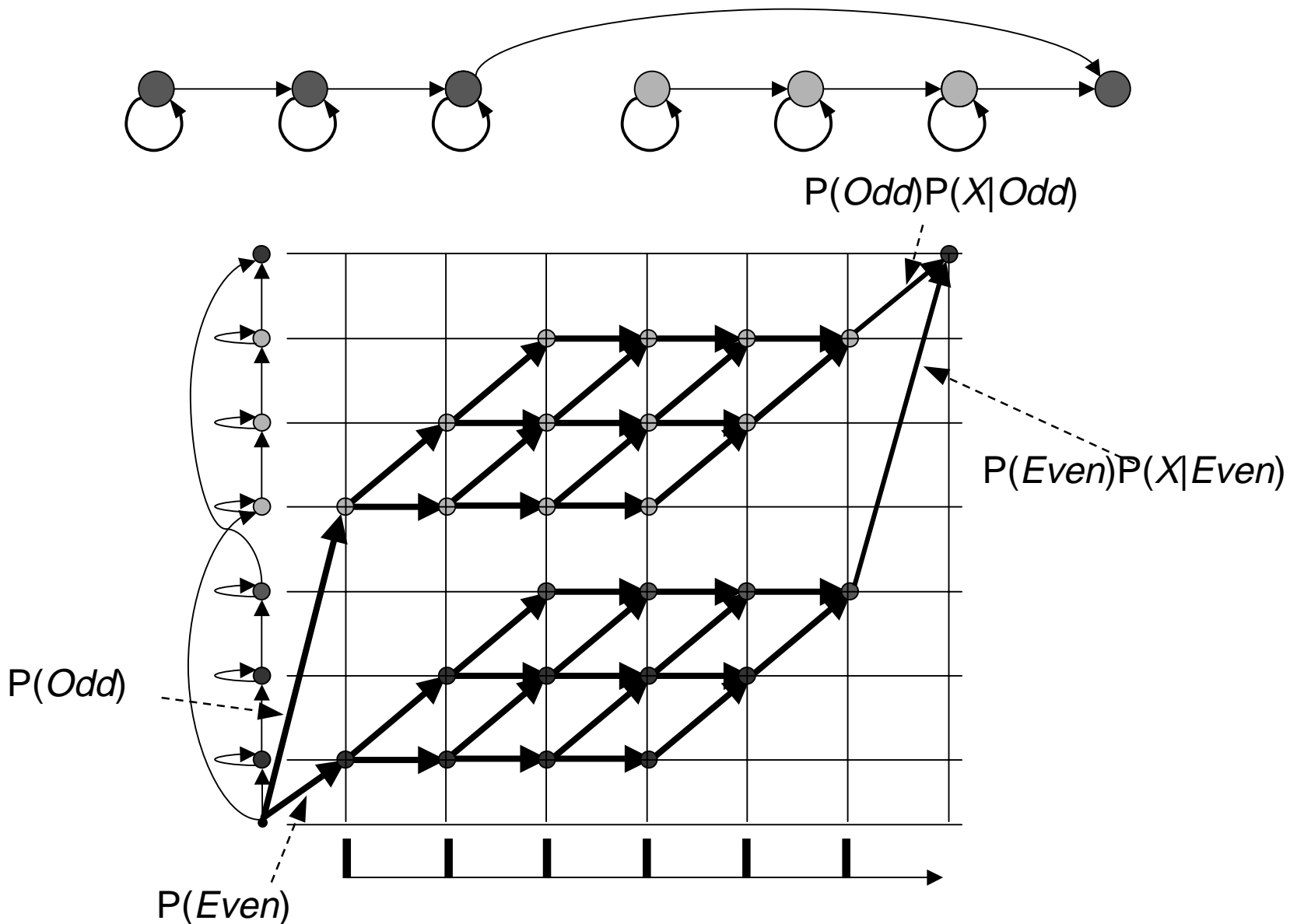


$$P(\text{Even})P(X|\text{Even})$$



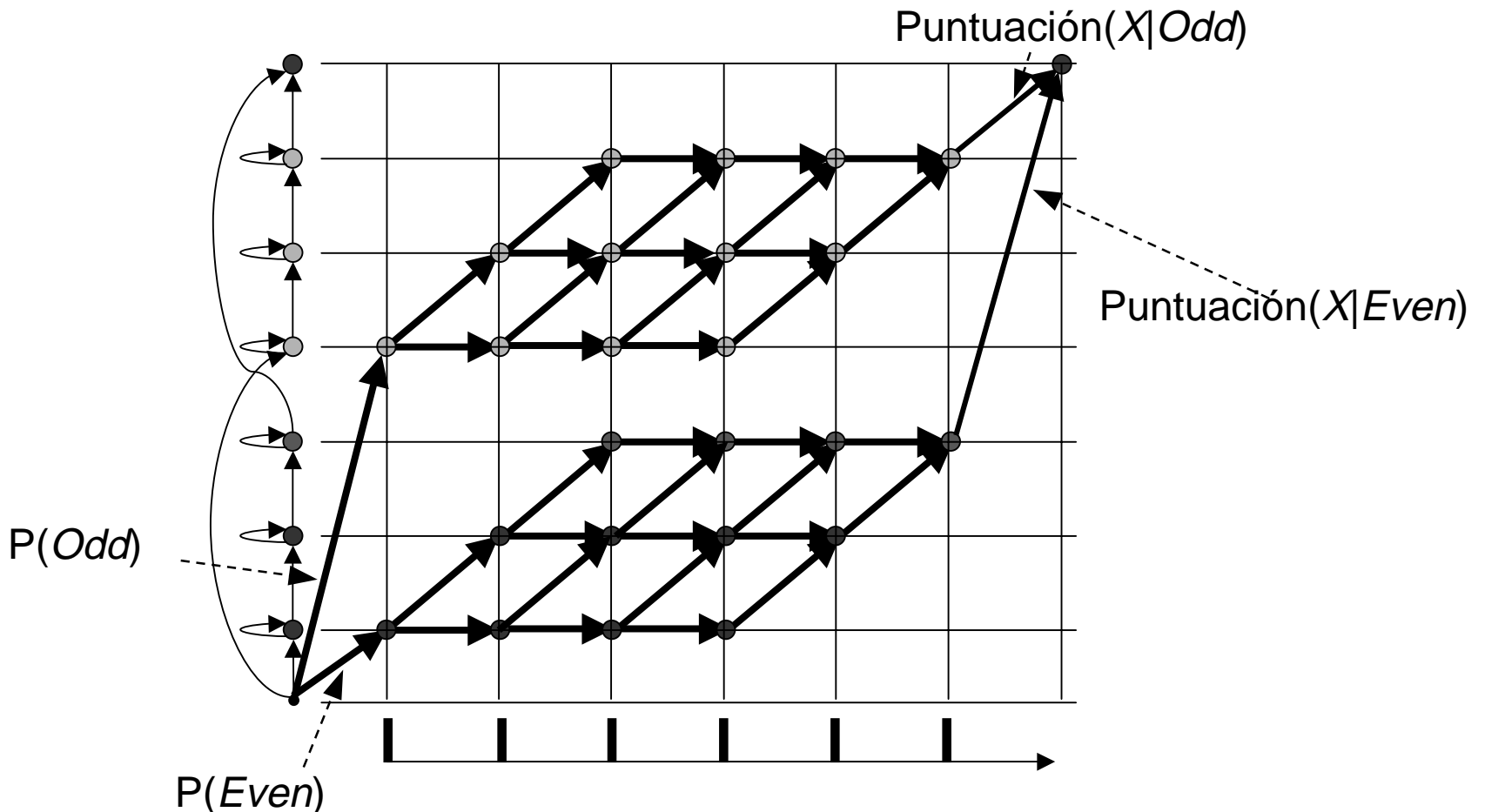
$$P(\text{Even})$$

Clasificación entre *Odd* y *Even*



Descodificación para la clasificación entre *Odd* y *Even*

- ◆ La probabilidad total aproximada de todas las rutas con probabilidad de la mejor ruta
 - Las computaciones pueden realizarse en el dominio logarítmico. Sólo son necesarias adiciones y comparaciones
 - Menos costoso que el de avance pleno donde las multiplicaciones y las adiciones son necesarias



Computación de las puntuaciones de la mejor ruta para la decodificación de Viterbi

- ◆ La puntuación aproximada para una palabra es:

$$P(X | word) \approx \max_{s \in \{\text{todas las secuencias de estado}\}} \{P(X, s | word)\}$$

- ◆ Escrito de forma explícita

$$P(X_1..X_t | word) \approx \max_{s_1, s_2, \dots, s_T} \{\pi(s_1)P(X_1 | s_1)P(s_2 | s_1)P(X_2 | s_2) \dots P(s_T | s_{T-1})P(X_T | s_T)\}$$

↓
secuencia de estado

- ◆ La puntuación de la palabra puede computarse recursivamente mediante el algoritmo de Viterbi

$$P_s(t) = \max_{s'} \{P_{s'}(t-1)P(s | s')P(X_t | s)\}$$

↓

Puntuación de la mejor ruta de todas las secuencias de estado que acaban en el estado s en tiempo t

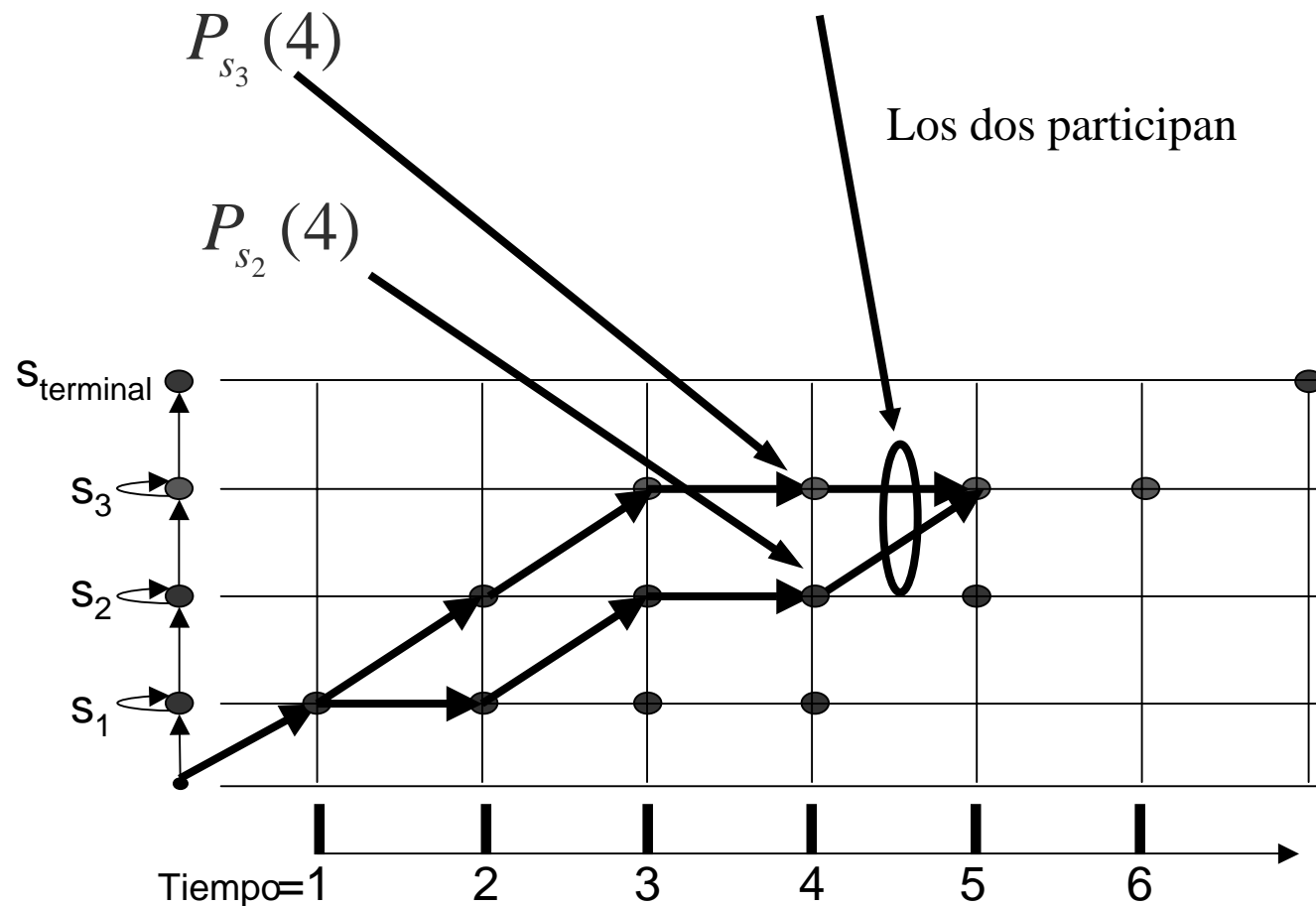
↓

Puntuación de la mejor ruta de todas las secuencias de estado que acaban en el estado s' en tiempo $t-1$

Computación de las puntuaciones de la mejor ruta para la descodificación de Viterbi

◆ La recursión hacia delante

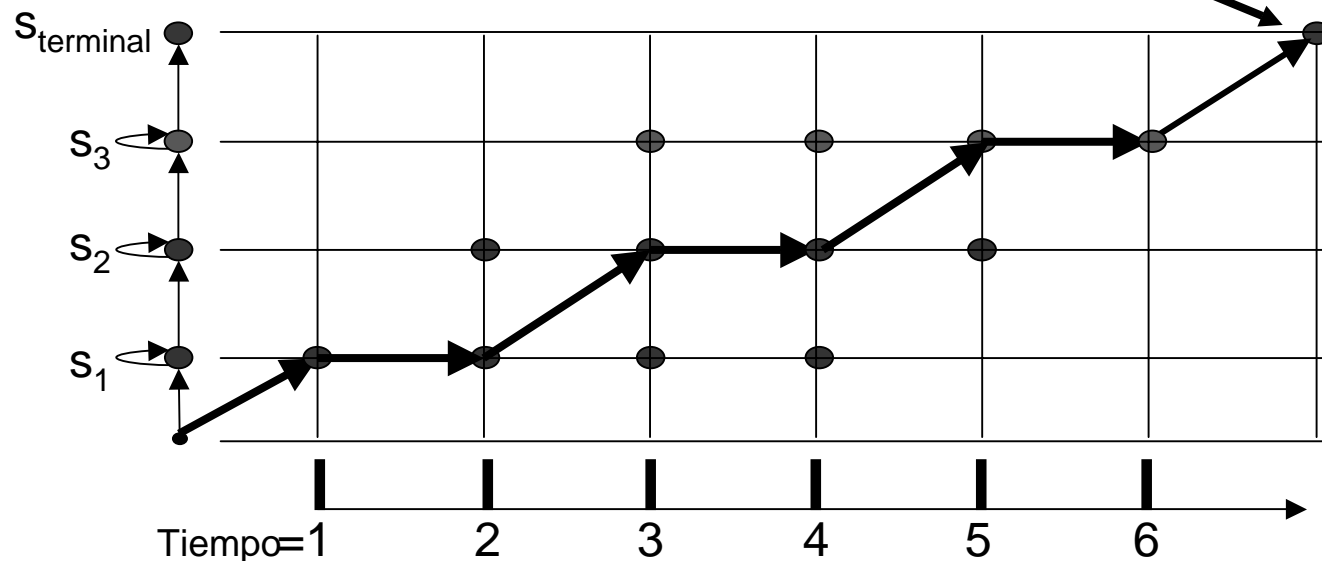
$$P_{s_3}(5) = \max\{ P_{s_3}(4)P(s_3 | s_3)P(X_5 | s_3), P_{s_2}(4)P(s_3 | s_2)P(X_5 | s_3) \}$$



Computación de las puntuaciones de la mejor ruta para la descodificación de Viterbi

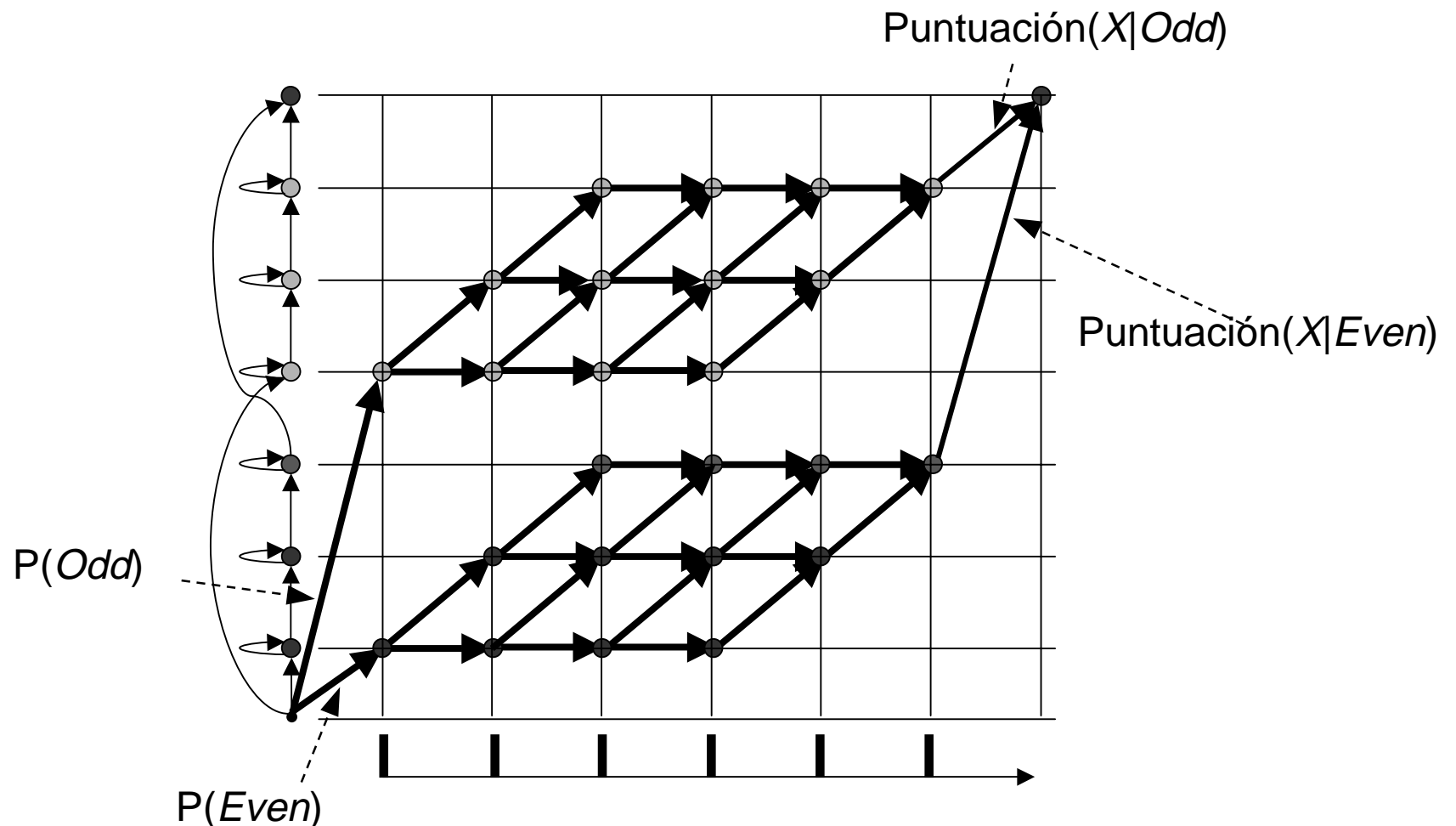
- ◆ La recursión hacia delante está calificada por la *descodificación* de Viterbi
 - La terminología proviene de la descodificación de los códigos de corrección del error
- ◆ La puntuación para la palabra es la puntuación de la ruta que sale adelante hasta el estado terminal
 - Empleamos el término *puntuación* para distinguirlo de la probabilidad total de la palabra

$$\text{Puntuación}(X | \text{word}) = P_{S_{\text{terminal}}}(T)$$



Descodificando para clasificar entre *Odd* y *Even*

- ◆ Compare las puntuaciones (mejores probabilidades de secuencia de estado) de todas las palabras participantes



Descodificación de secuencias de palabras

Clasificación estadística de secuencias de palabras

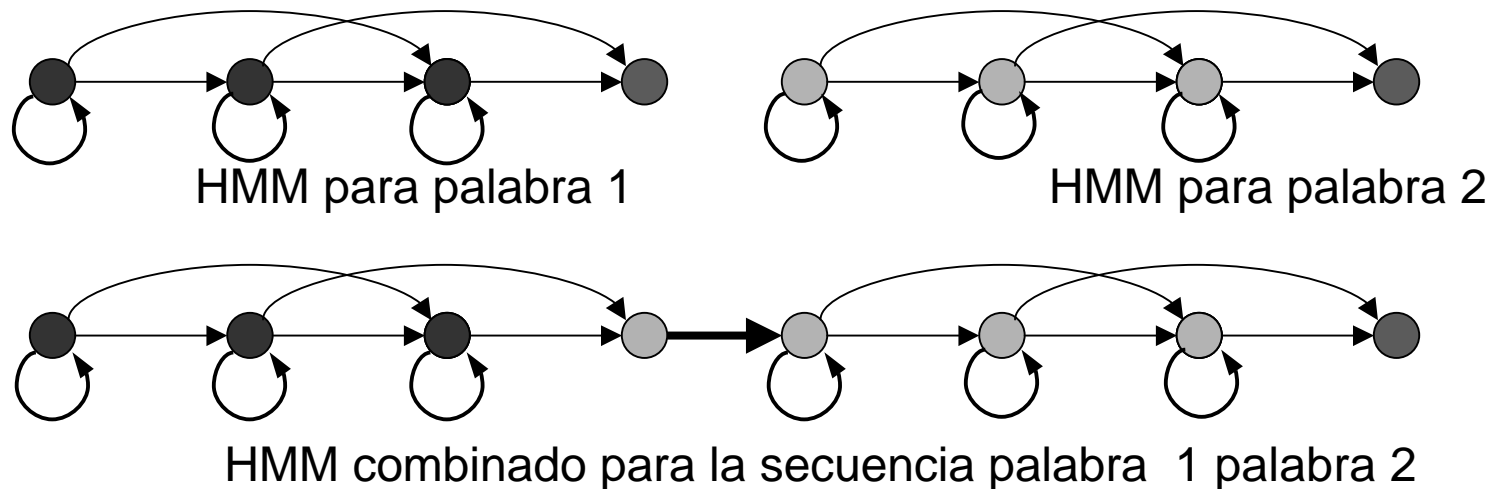
- ◆ Las clases son secuencias de palabras
- ◆ Los datos son grabaciones habladas de secuencias de palabras
- ◆ Clasificación bayesiana:

$$\text{word}_1, \text{word}_2, \dots, \text{word}_N = \arg \max_{wd_1, wd_2, \dots, wd_N} \{P(X | wd_1, wd_2, \dots, wd_N)P(wd_1, wd_2, \dots, wd_N)\}$$

- ◆ $P(wd_1, wd_2, wd_3..)$ es una probabilidad *a priori* de la secuencia de palabra $wd_1, wd_2, wd_3..$
 - Obtenida a partir de un modelo del lenguaje
- ◆ $P(X | wd_1, wd_2, wd_3..)$ es la probabilidad de X computada sobre la función de distribución de probabilidad de la secuencias de la palabra $wd_1, wd_2, wd_3..$
 - Los HMM representan ahora las distribuciones de probabilidad de las secuencias de palabra

Construyendo un HMM para secuencias de palabras

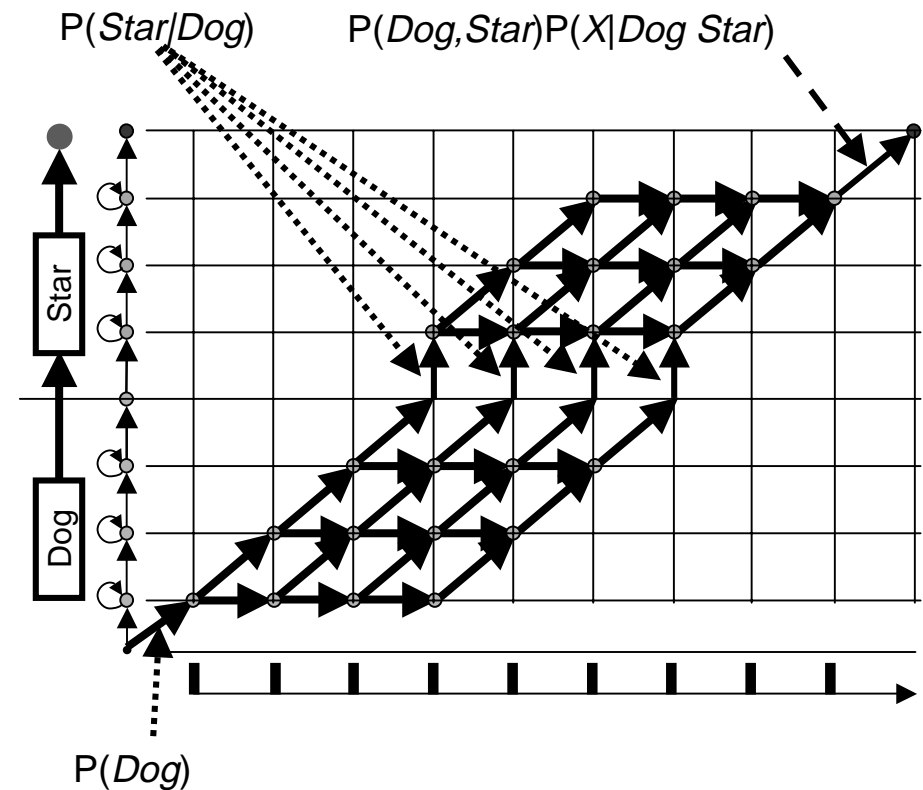
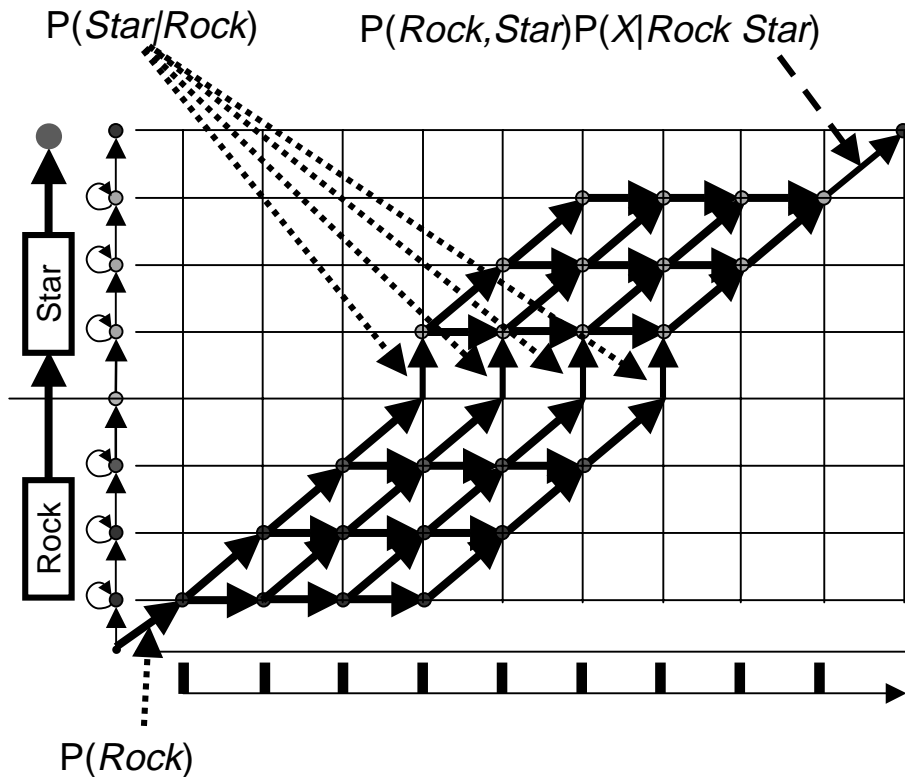
- ◆ Concatenar los HMM para cada una de las palabras de la secuencia



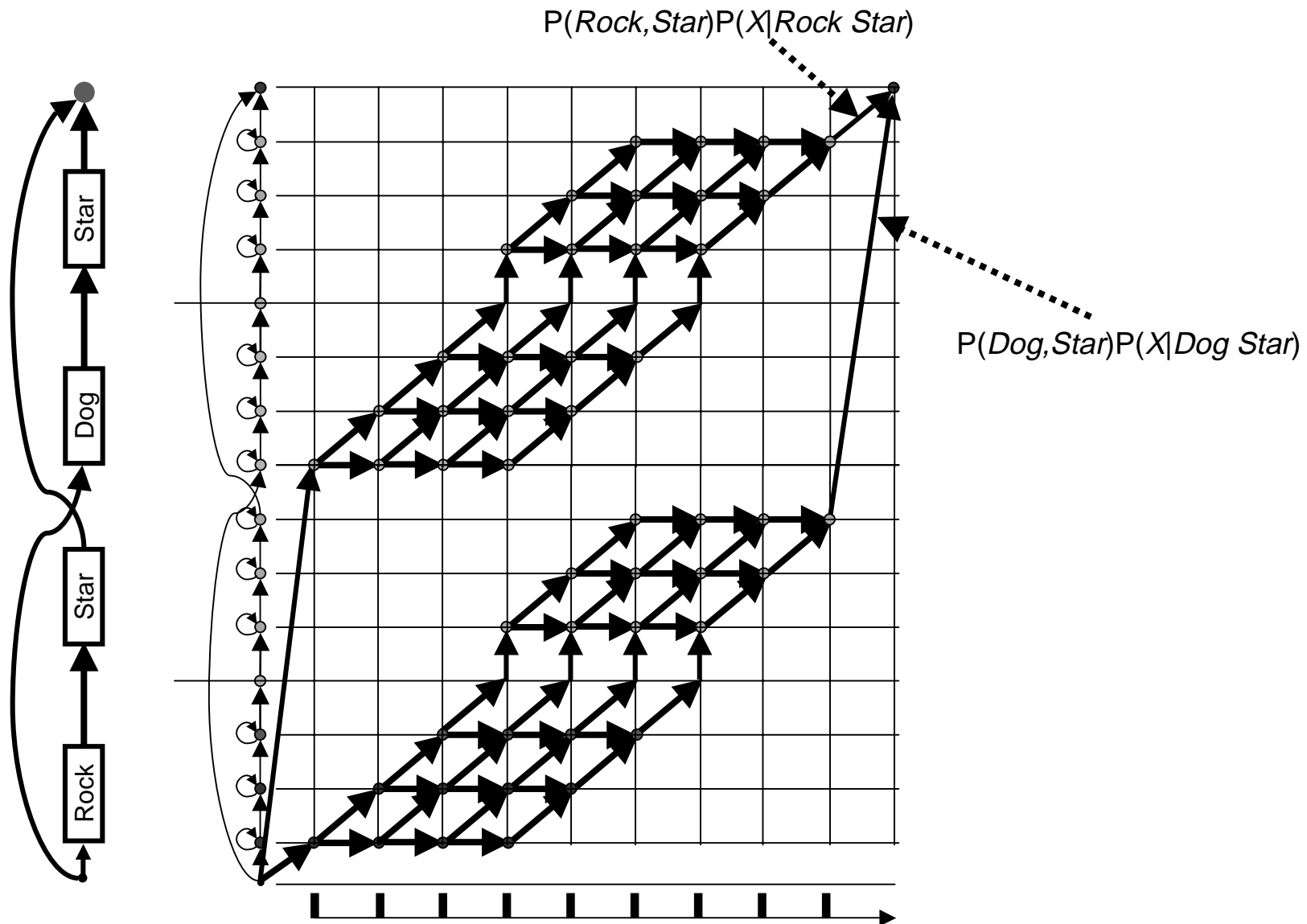
- ◆ De hecho, los mismos HMM de la palabra se construyen a menudo al concatenar los HMM para los fonemas
 - Existen bastantes menos fonemas que palabras, y aparecen con mayor frecuencia en los datos de entrenamiento
 - Las palabras que nunca se encuentran en los datos de entrenamiento se pueden construir a partir de los HMM del fonema

Clasificación bayesiana entre secuencias de palabras

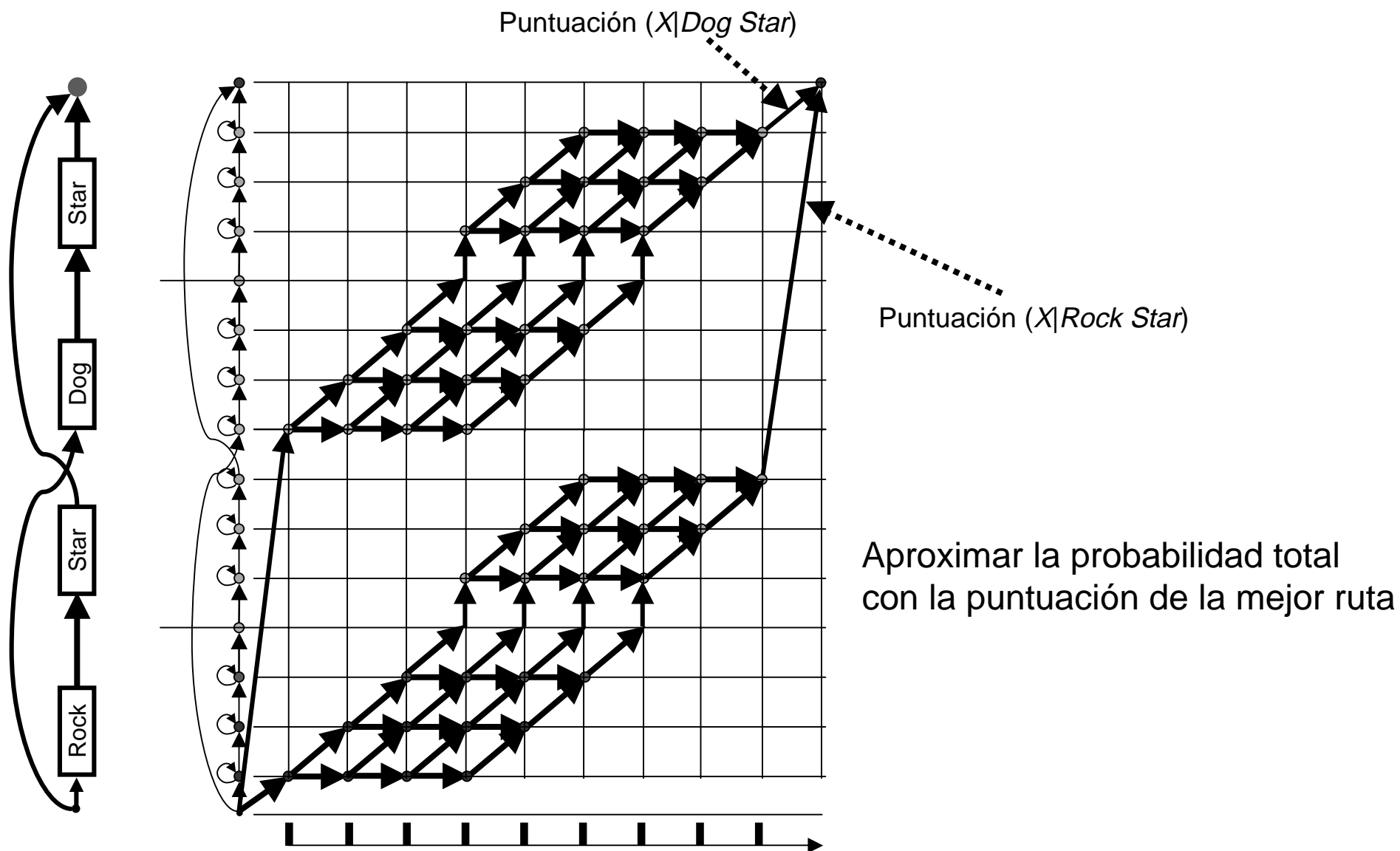
- ◆ Clasificación de un enunciado como “*Rock Star*” o “*Dog Star*”
 - ◆ Debe compararse $P(\text{Rock}, \text{Star})P(X|\text{Rock Star})$ con $P(\text{Dog}, \text{Star})P(X|\text{Dog Star})$



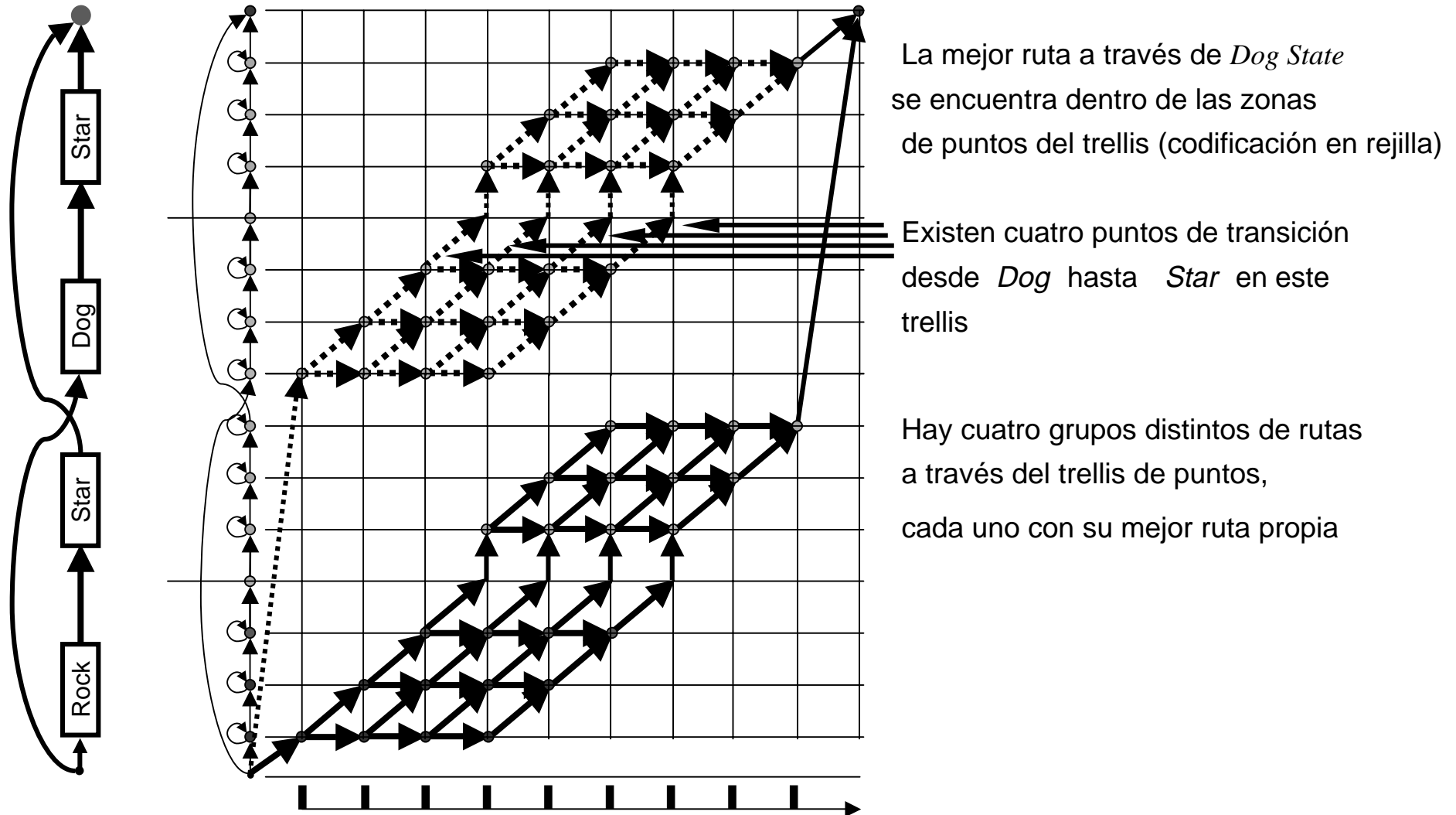
Clasificación bayesiana entre secuencias de palabras



Descodificación de secuencias de palabras

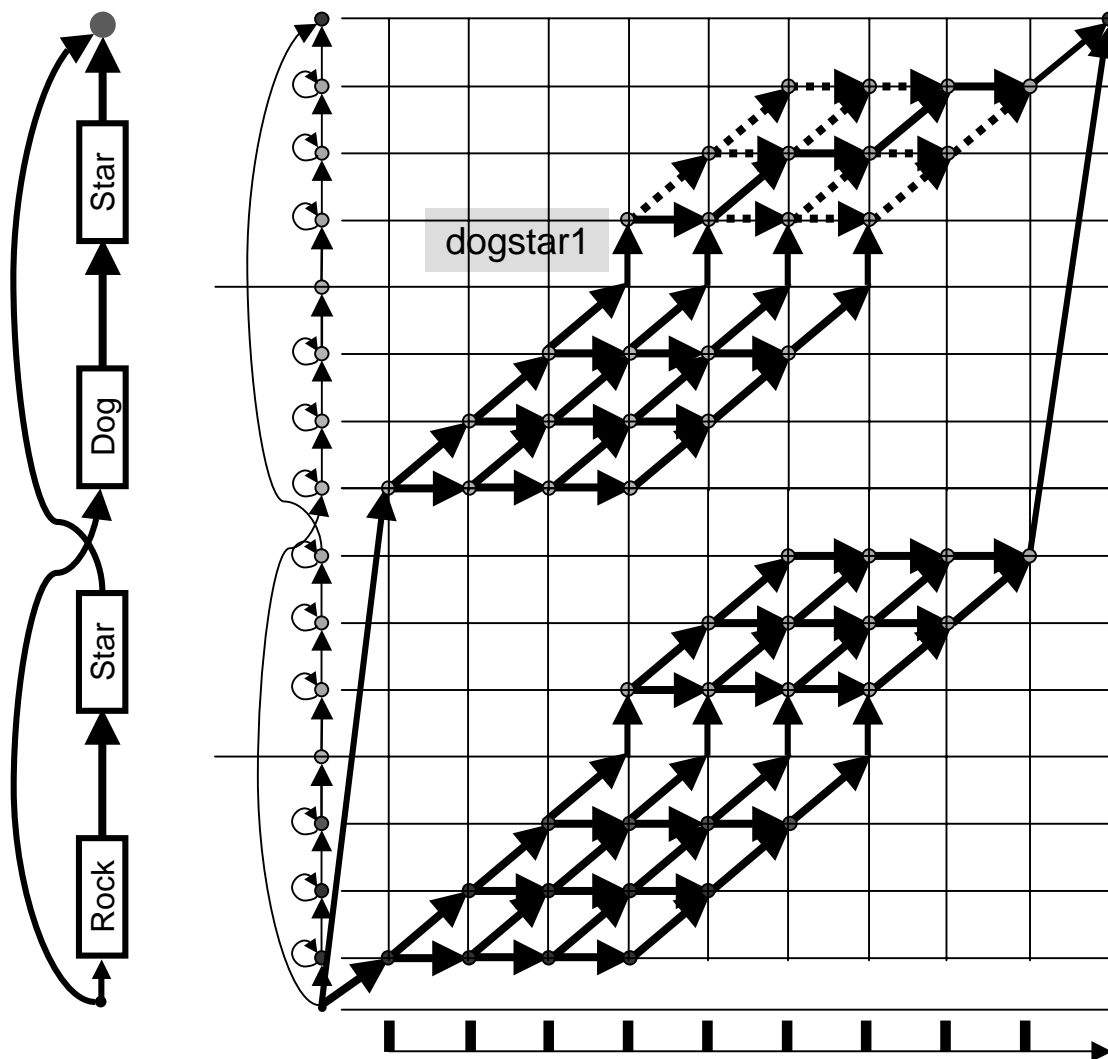


Descodificación de las secuencias de palabras



Descodificación de secuencias de palabras

GRUPO 1 y su mejor ruta



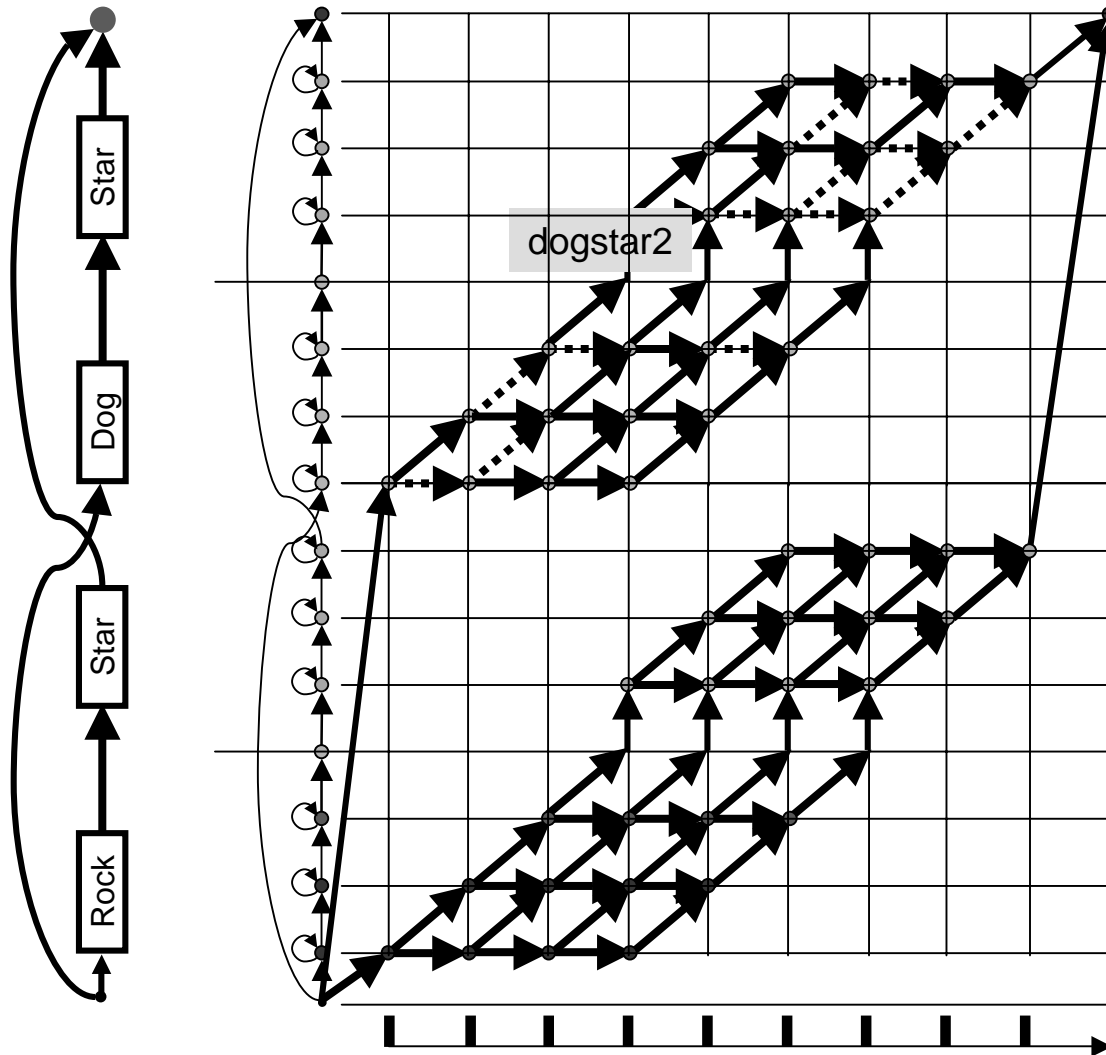
La mejor ruta a través de *Dog Star* se encuentra dentro de las zonas de puntos del diagrama de trellis

Existen cuatro puntos de transición desde *Dog* hasta o *Star* en este trellis

Existen cuatro grupos distintos de rutas a través del trellis de puntos, cada uno con su mejor ruta propia

Descodificación de secuencias de palabras

GRUPO 2 y su mejor ruta



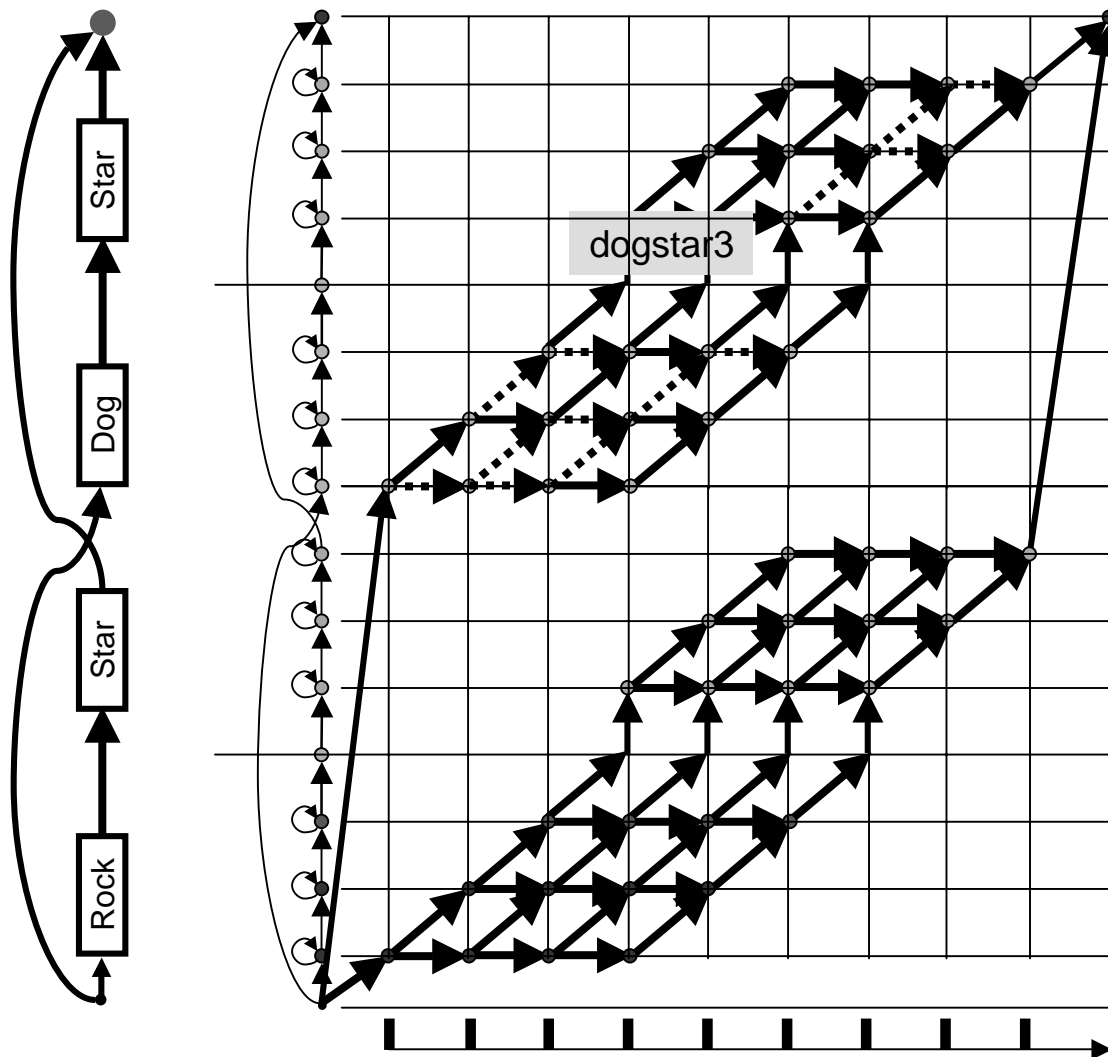
La mejor ruta a través de Dog Star se encuentra dentro de las zonas de puntos del trellis

Existen cuatro puntos de transición desde *Dog* hasta *Star* en este trellis

Existen cuatro grupos de rutas distintos a través del trellis de puntos, cada uno con su mejor ruta propia

Descodificación de secuencias de palabras

CONJUNTO 3 y su mejor ruta



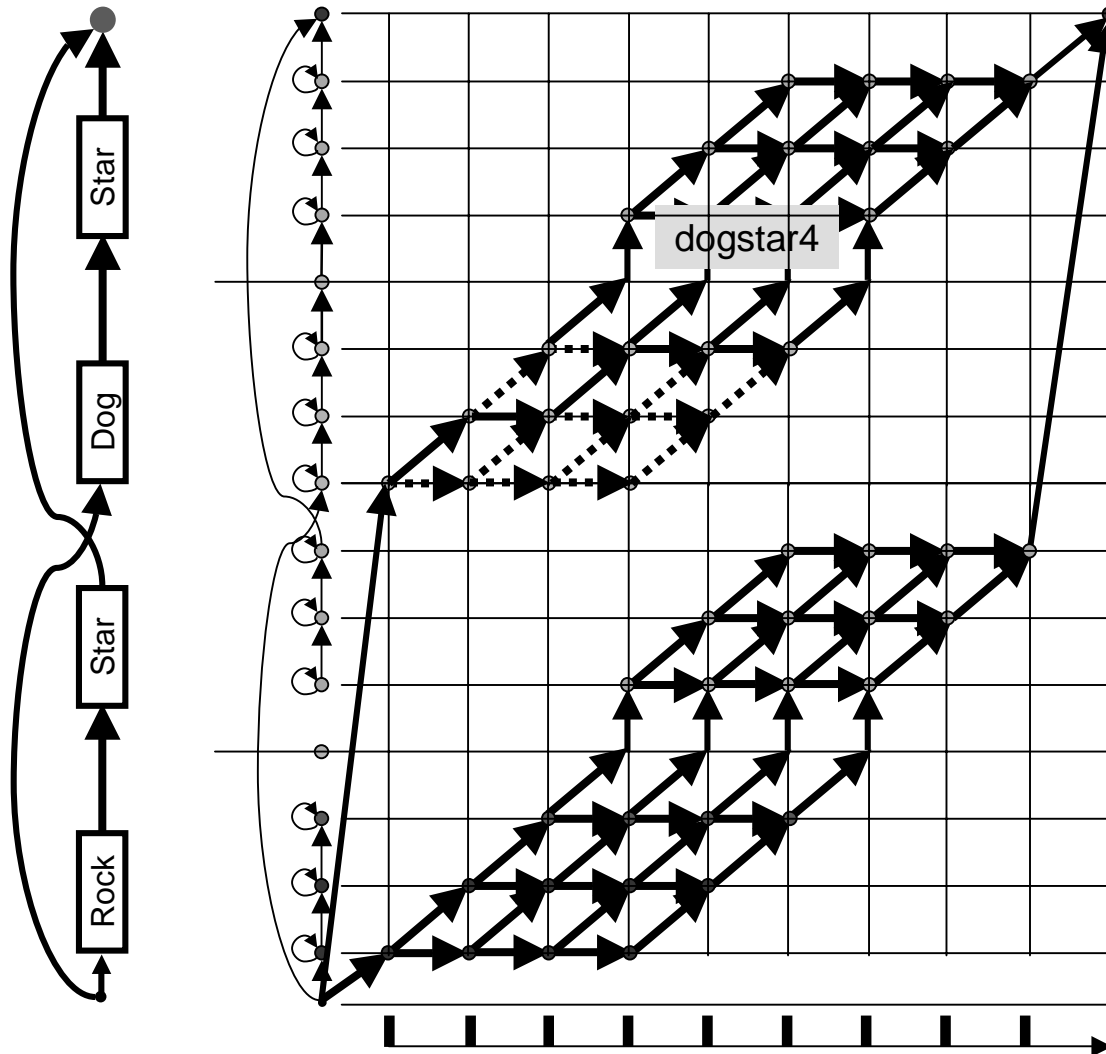
La mejor ruta a través de *Dog Star* se encuentra dentro de las zonas de puntos del trellis

Existen cuatro puntos de transición desde Dog hasta Star en este trellis

Existen cuatro grupos distintos de rutas a través del trellis de puntos, cada uno con su mejor ruta propia

Decodificación de secuencias de palabras

GRUPO 4 y su mejor ruta

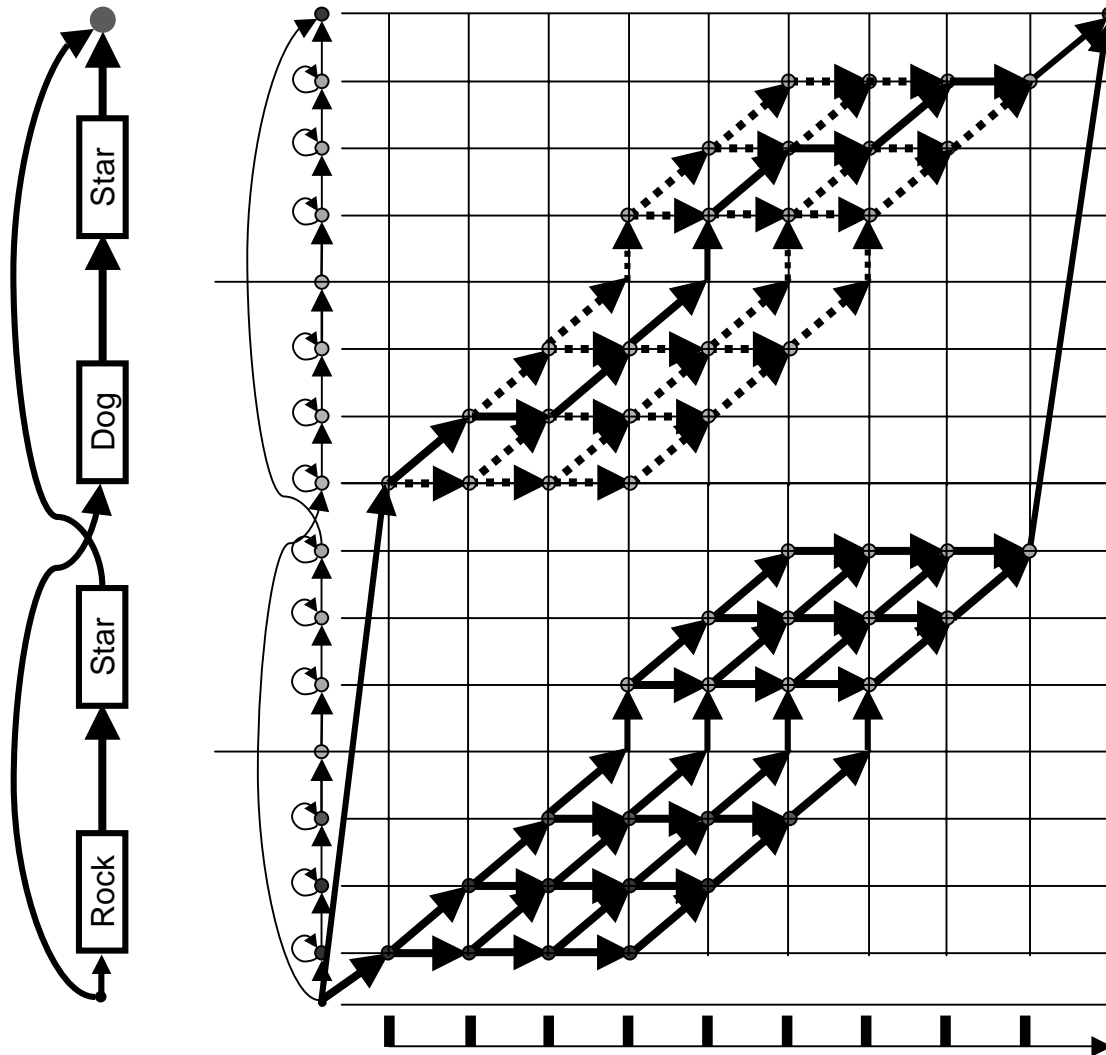


La mejor ruta a través de *Dog Star* le encuentra dentro de las zonas de puntos del trellis

Existen cuatro puntos de transición desde *Dog* hasta *Star* en este trellis

Existen cuatro grupos distintos de rutas a través del trellis de puntos, cada uno con su mejor ruta propia

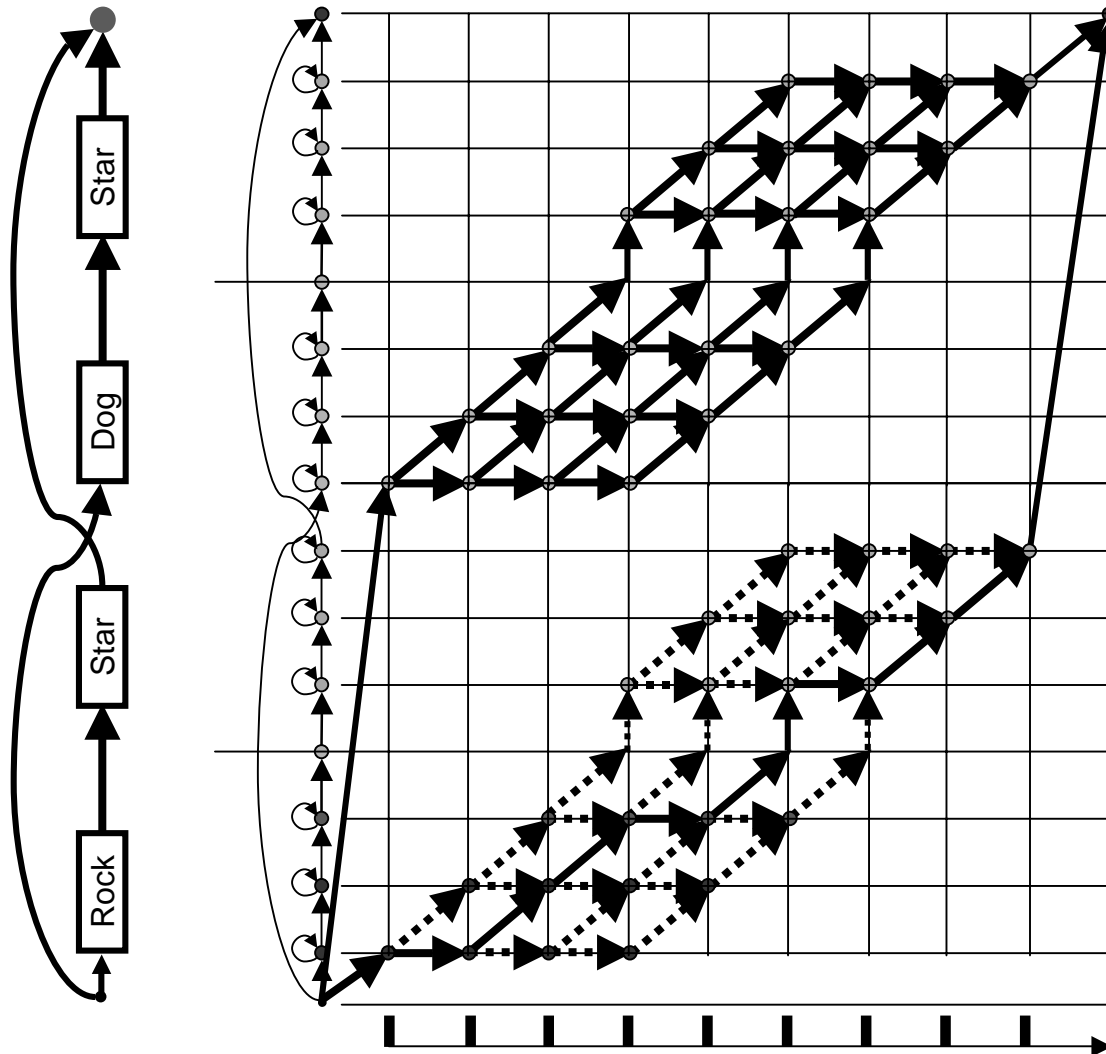
Decodificación de secuencias de palabras



La mejor ruta a través de *Dog Star* es la mejor de las cuatro mejores rutas propias de la transición

$$\max(\text{dogstar}) = \max(\text{dogstar1}, \text{dogstar2}, \text{dogstar3}, \text{dogstar4})$$

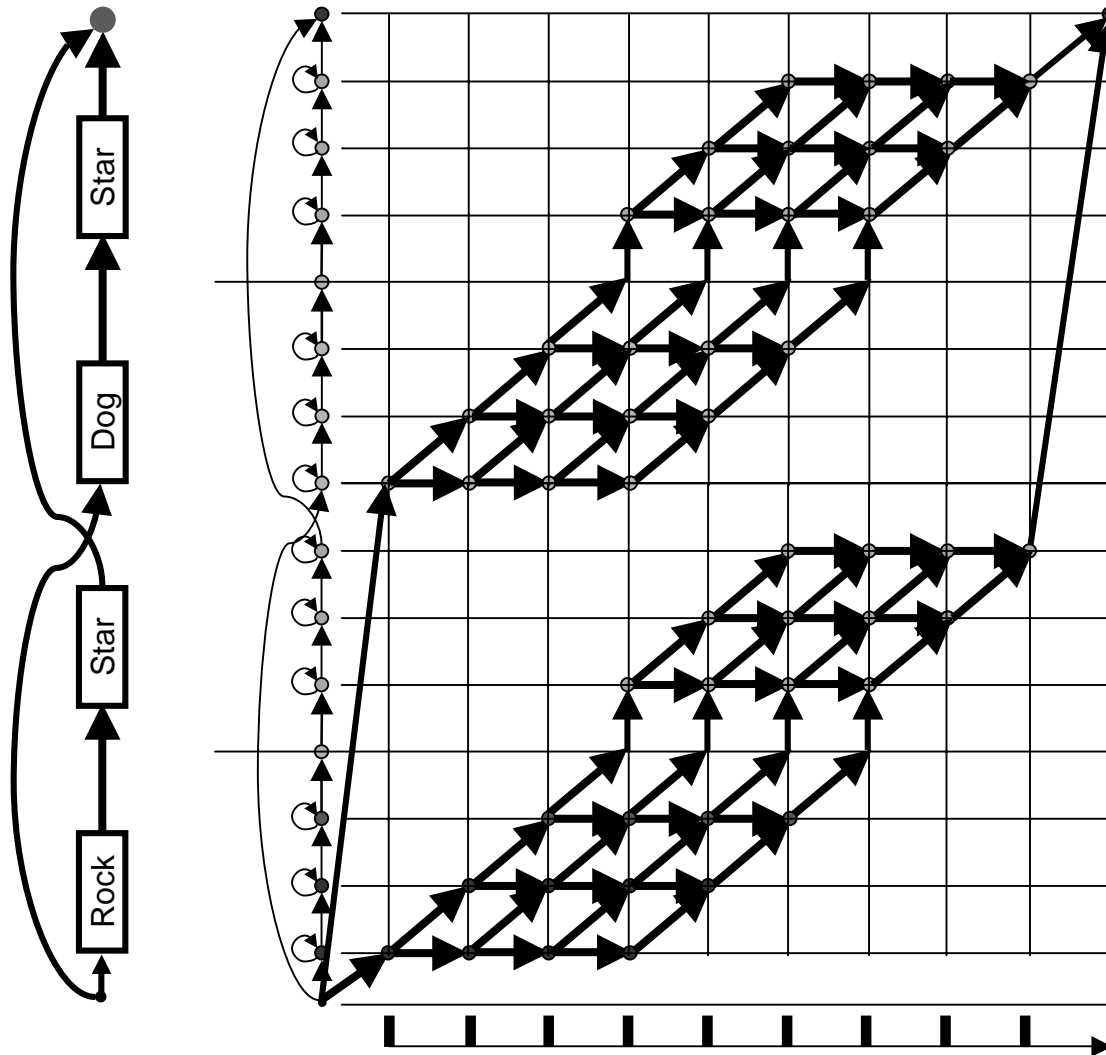
Decodificación de secuencias de palabras



De igual modo, para *Rock Star*, la mejor ruta a través del trellis es la mejor de las cuatro mejores rutas propias de la transición

$$\max(\text{rockstar}) = \max(\text{rockstar1}, \text{rockstar2}, \text{rockstar3}, \text{rockstar4})$$

Decodificación de secuencias de palabras



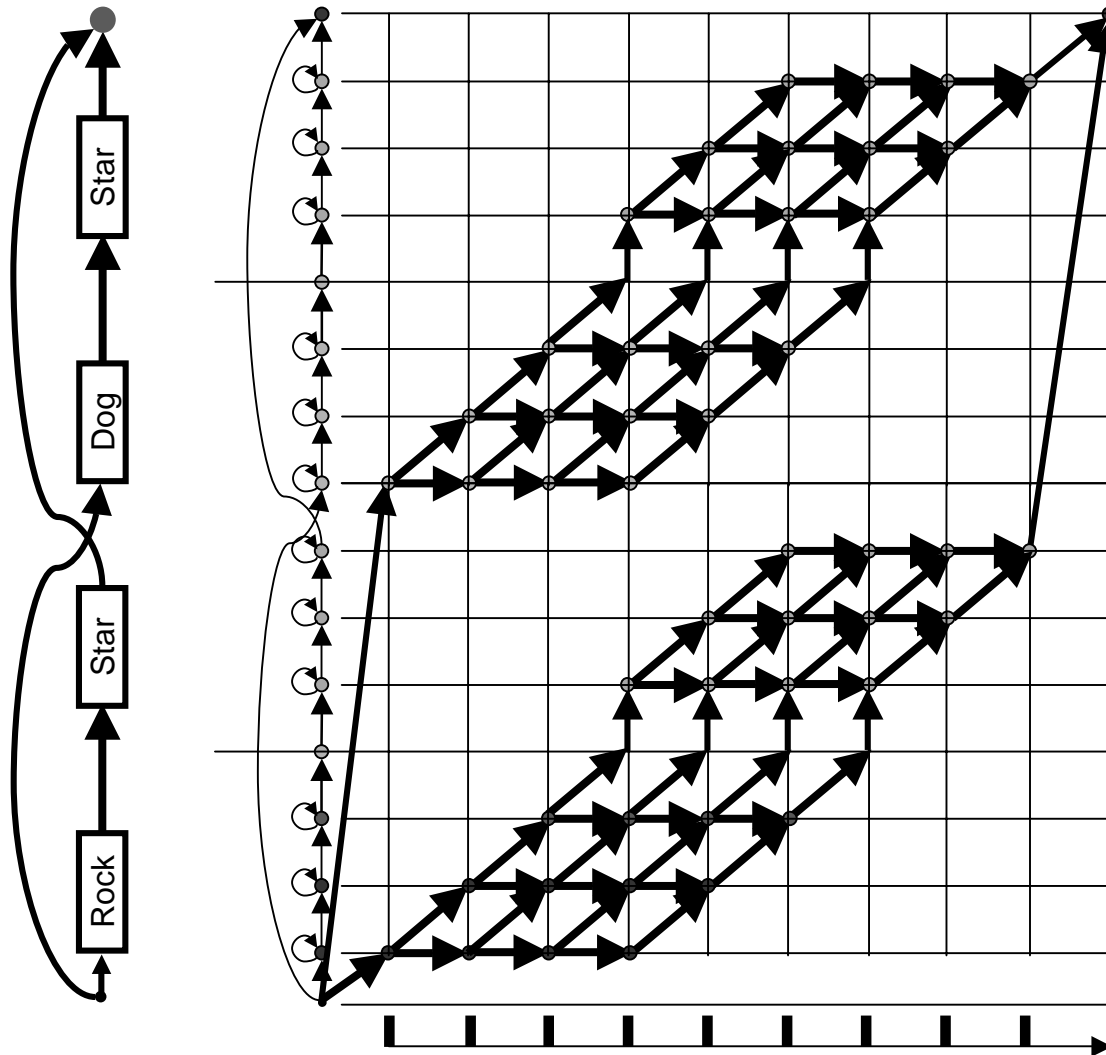
Luego compararíamos las mejores rutas a través de *Dog Star* y *Rock Star*

$\max(\text{dogstar}) =$
 $\max(\text{dogstar1}, \text{dogstar2},$
 $\text{dogstar3}, \text{dogstar4})$

$\max(\text{rockstar}) =$
 $\max(\text{rockstar1}, \text{rockstar2},$
 $\text{rockstar3}, \text{rockstar4})$

Viterbi =
 $\max(\max(\text{dogstar}),$
 $\max(\text{rockstar}))$

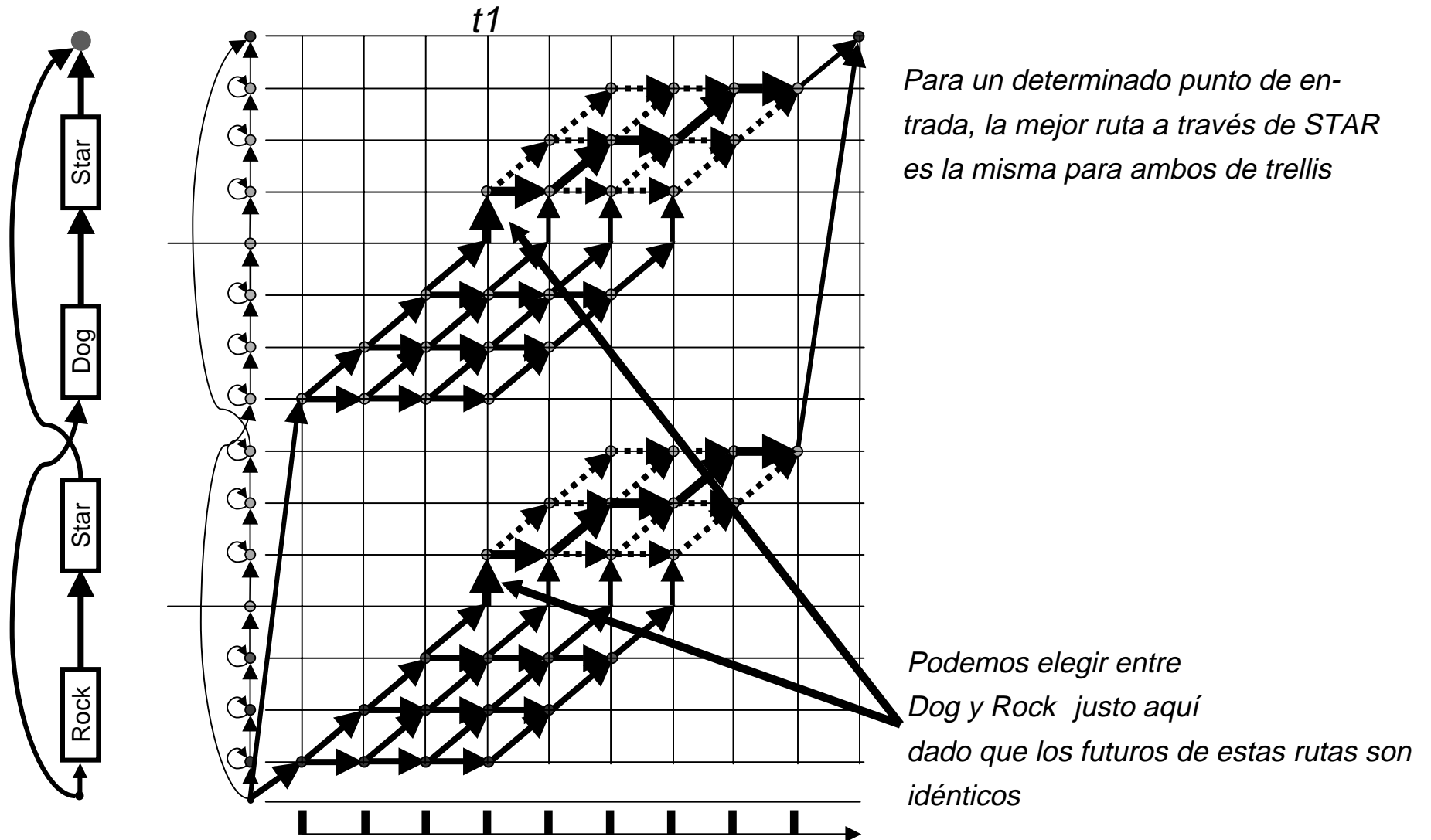
Decodificación de secuencias de palabras



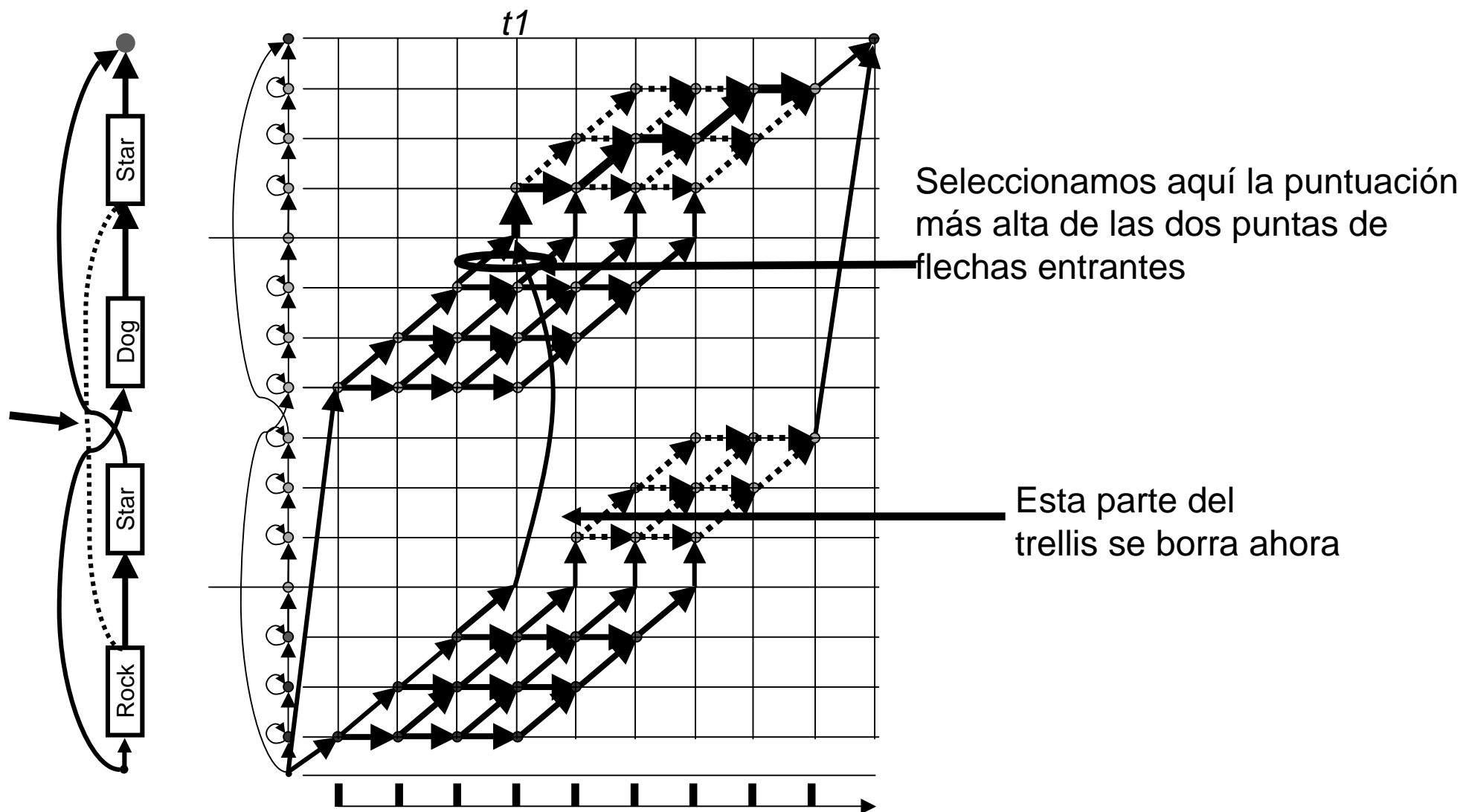
argmax es conmutativo:

$$\begin{aligned} & \max(\max(\text{dogstar}), \max(\text{rockstar})) \\ & = \\ & \max (\\ & \quad \max(\text{dogstar1}, \text{rockstar1}), \\ & \quad \max(\text{dogstar2}, \text{rockstar2}), \\ & \quad \max(\text{dogstar3}, \text{rockstar3}), \\ & \quad \max(\text{dogstar4}, \text{rockstar4}) \\ &) \end{aligned}$$

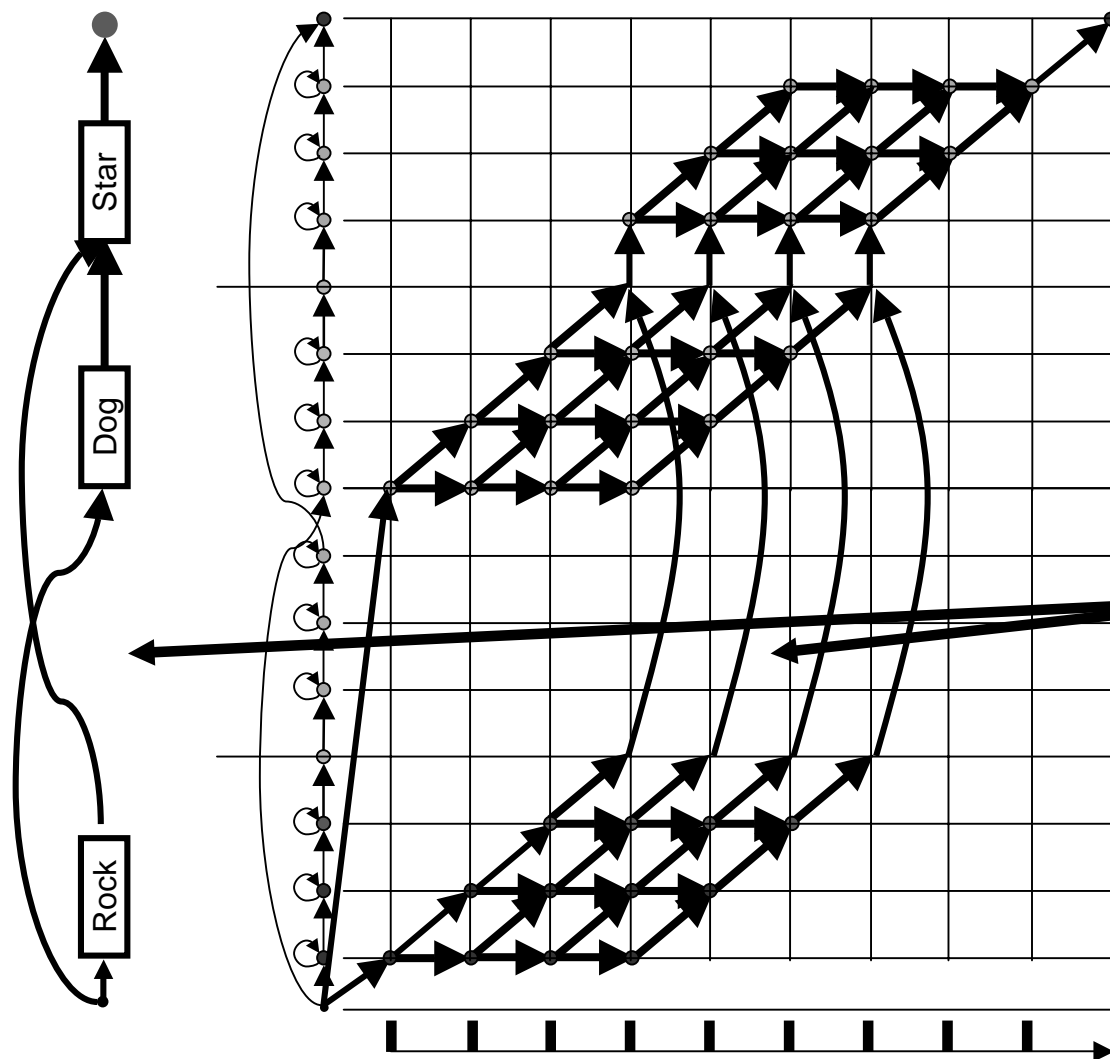
Decodificación de secuencias de palabras



Decodificación de secuencias de palabras



Decodificación de las secuencias de palabras

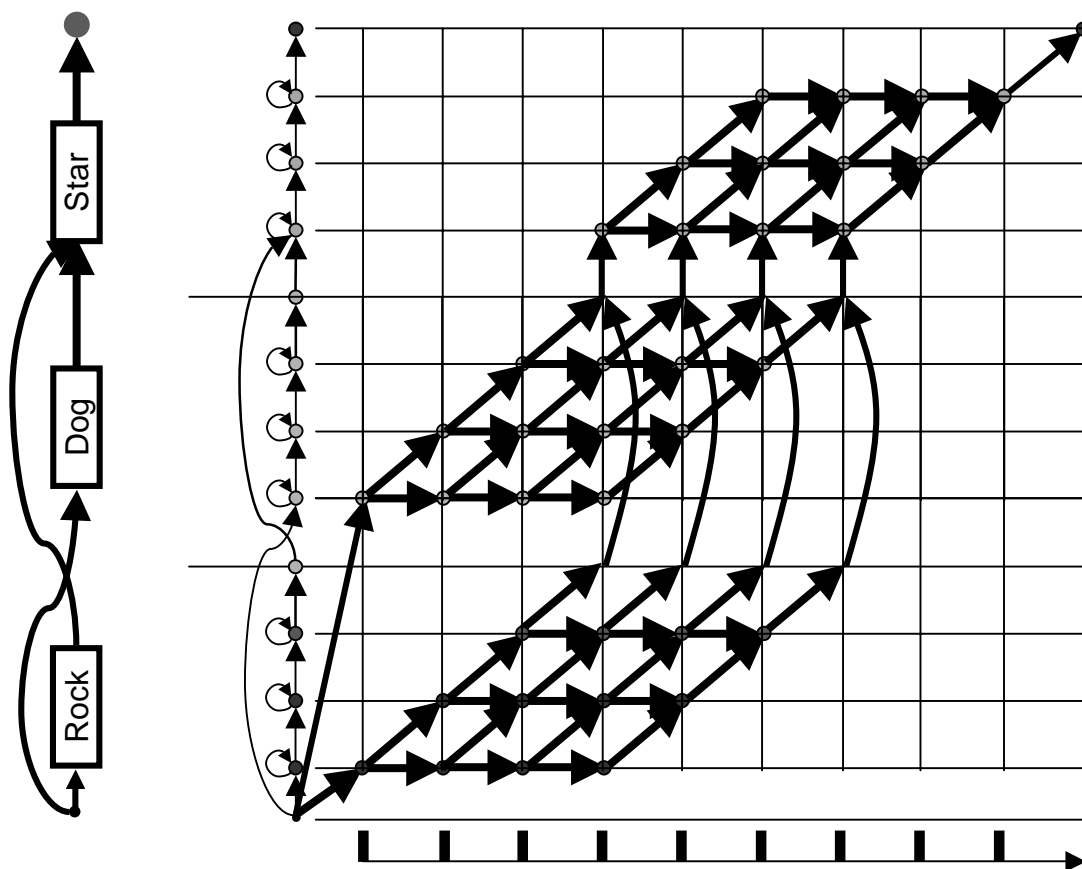


Una lógica parecida puede aplicarse en otros puntos de entrada a *Star*

Esta copia del trellis para *STAR* se borra totalmente

Descodificación de secuencias de palabras

- ◆ Los dos ejemplos de *Star* pueden plegarse en uno para formar un trellis más pequeño
 - Esto es posible porque la descodificación está basada en la puntuación de la mejor ruta, en vez de en una puntuación de avance total



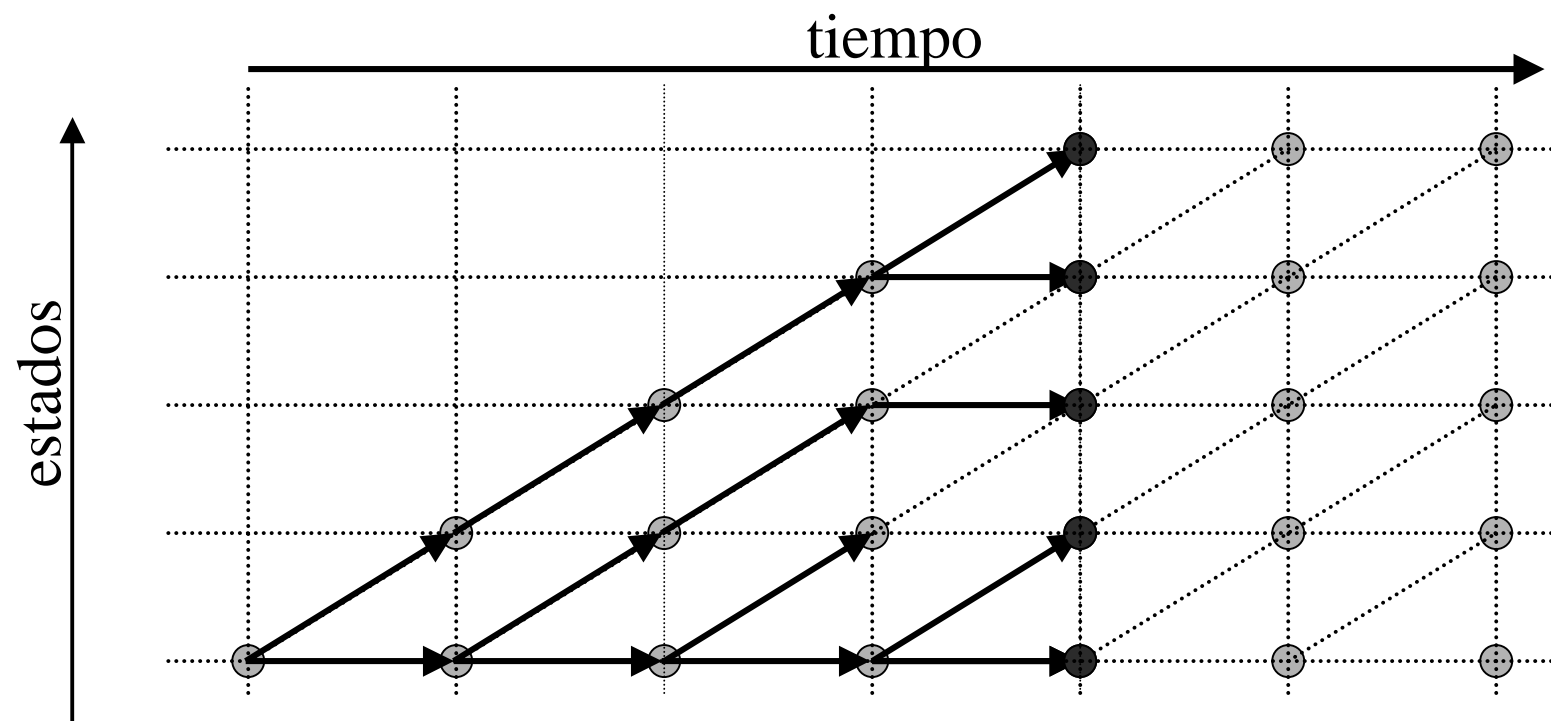
Sólo podemos llevar a cabo la descodificación de Viterbi en este gráfico plegado

La decodificación de avance total ya no es posible

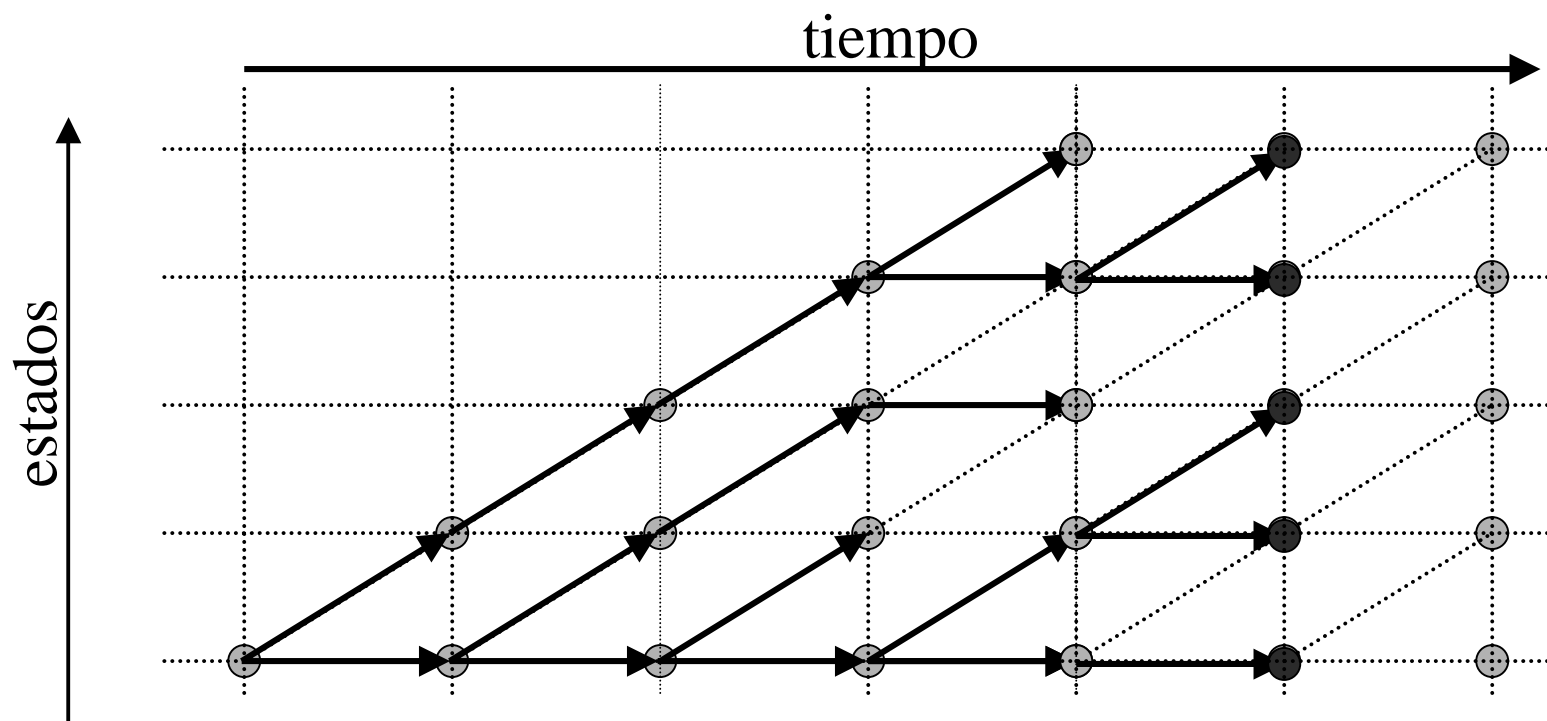
Ampliación de rutas a través del trellis:

Primera búsqueda en amplitud frente a primera en profundidad

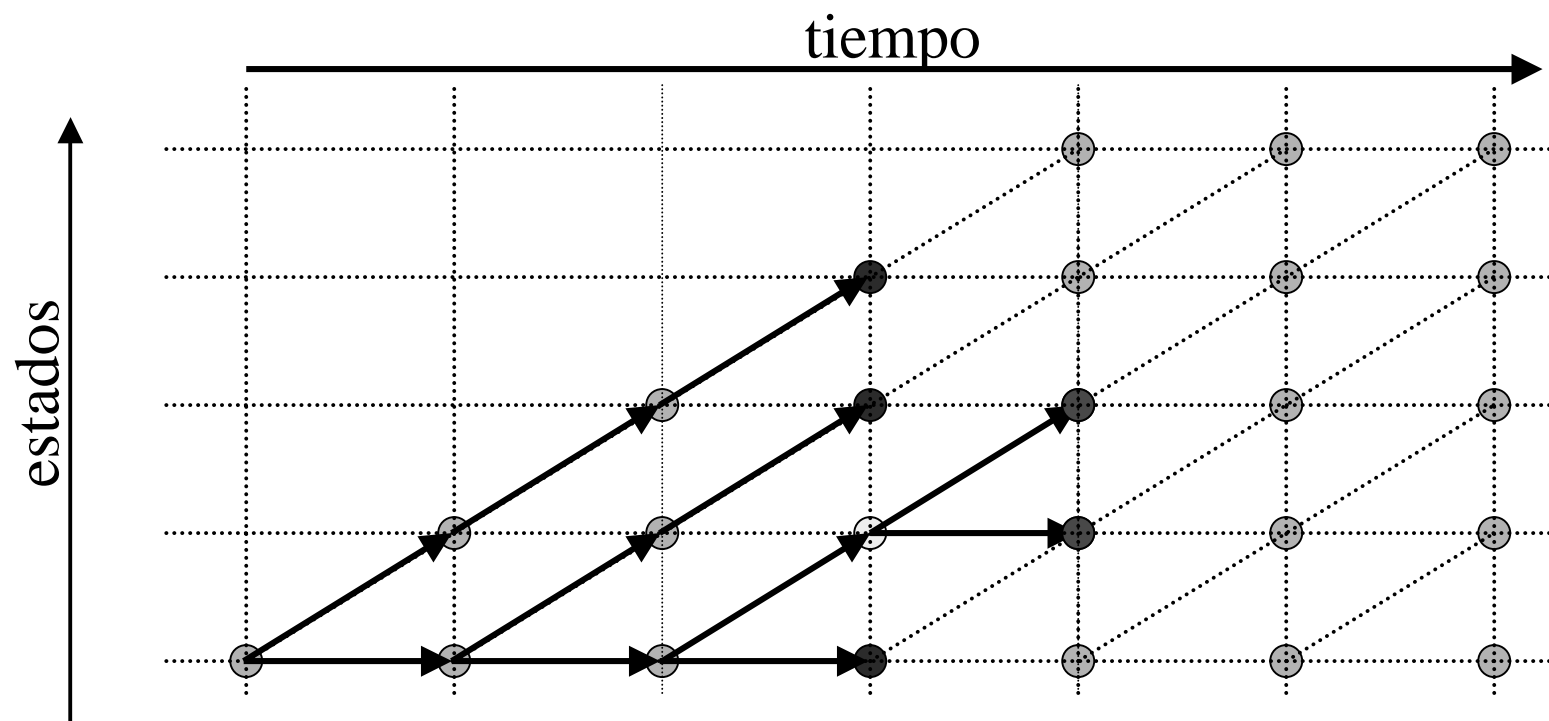
Búsqueda primero en amplitud



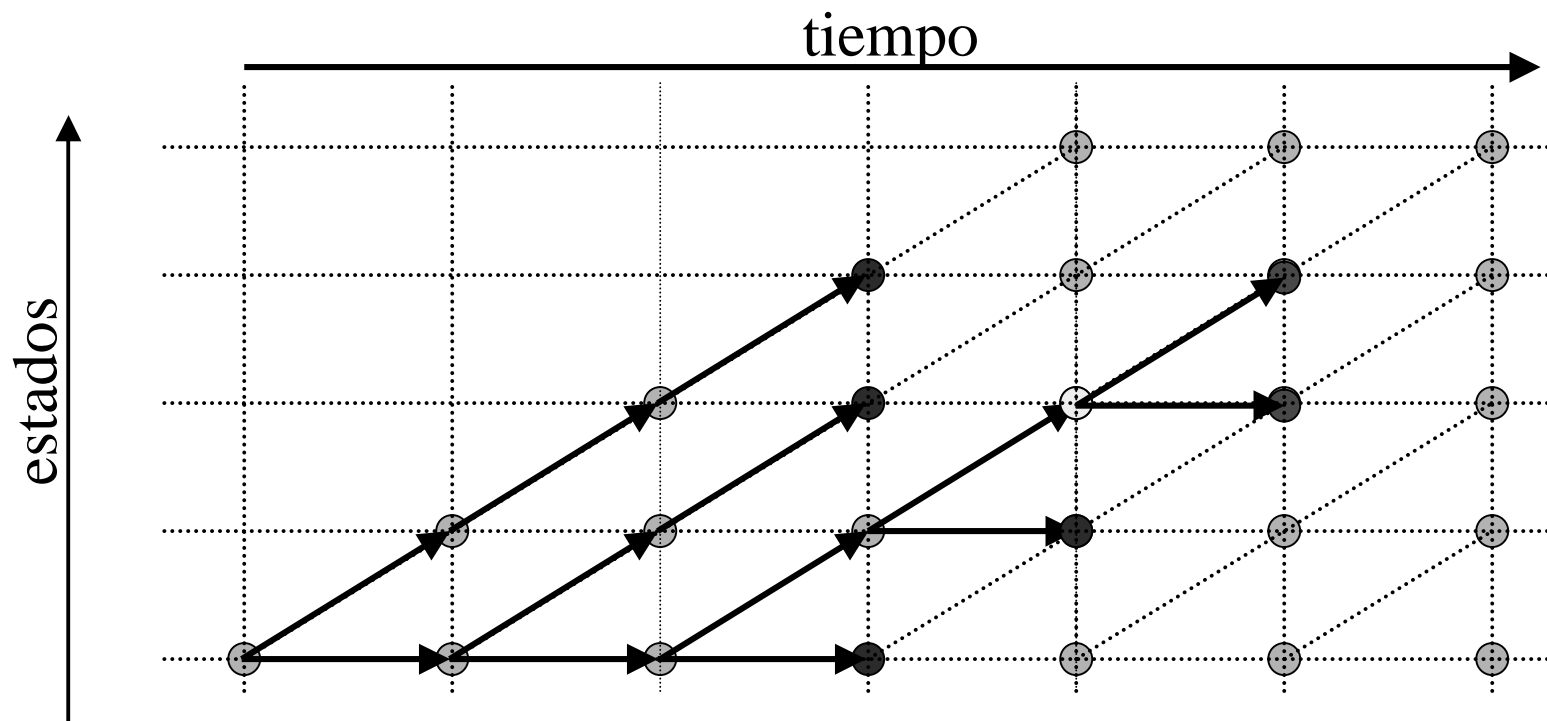
Búsqueda primero en amplitud



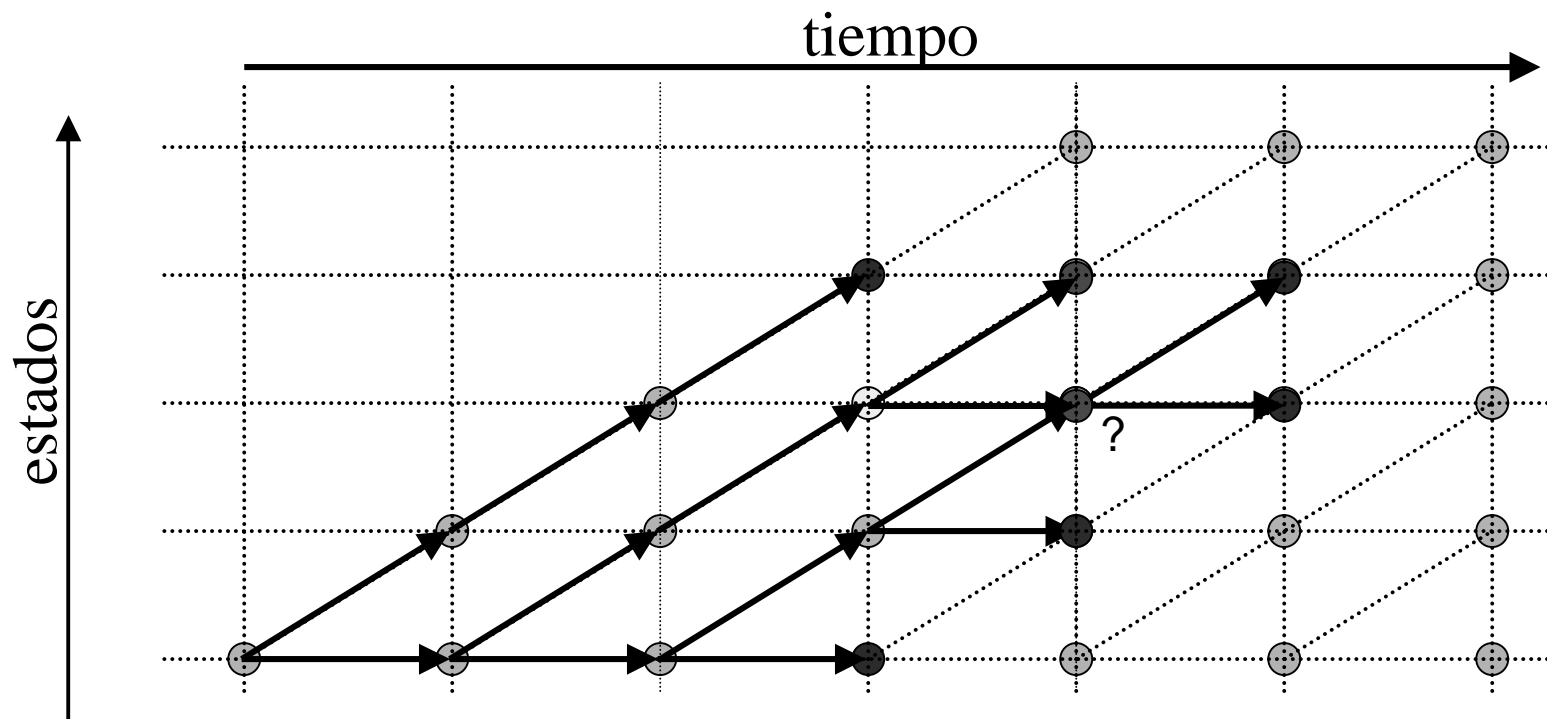
Búsqueda primero en profundidad



Búsqueda primero en profundidad



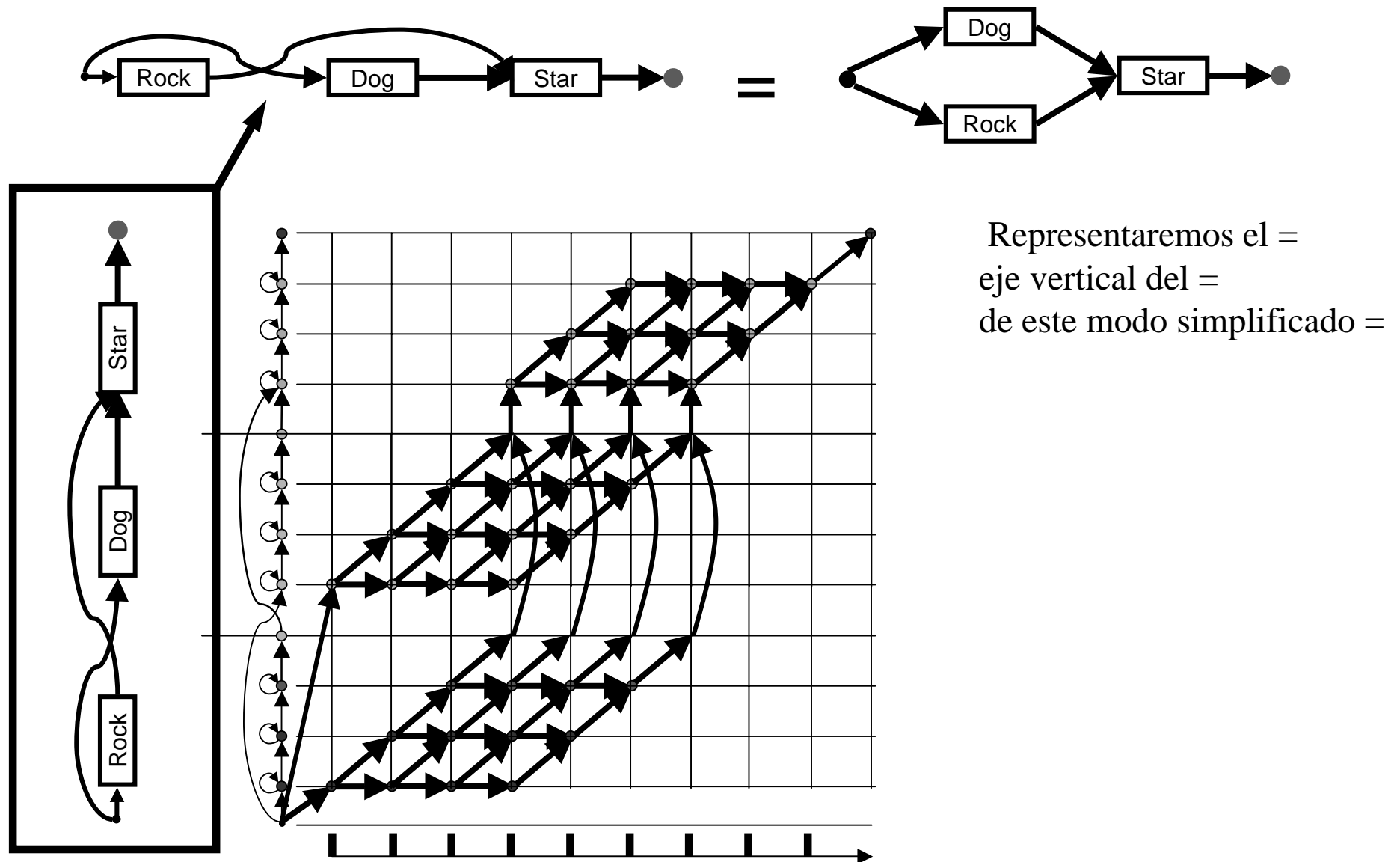
Búsqueda primero en profundidad



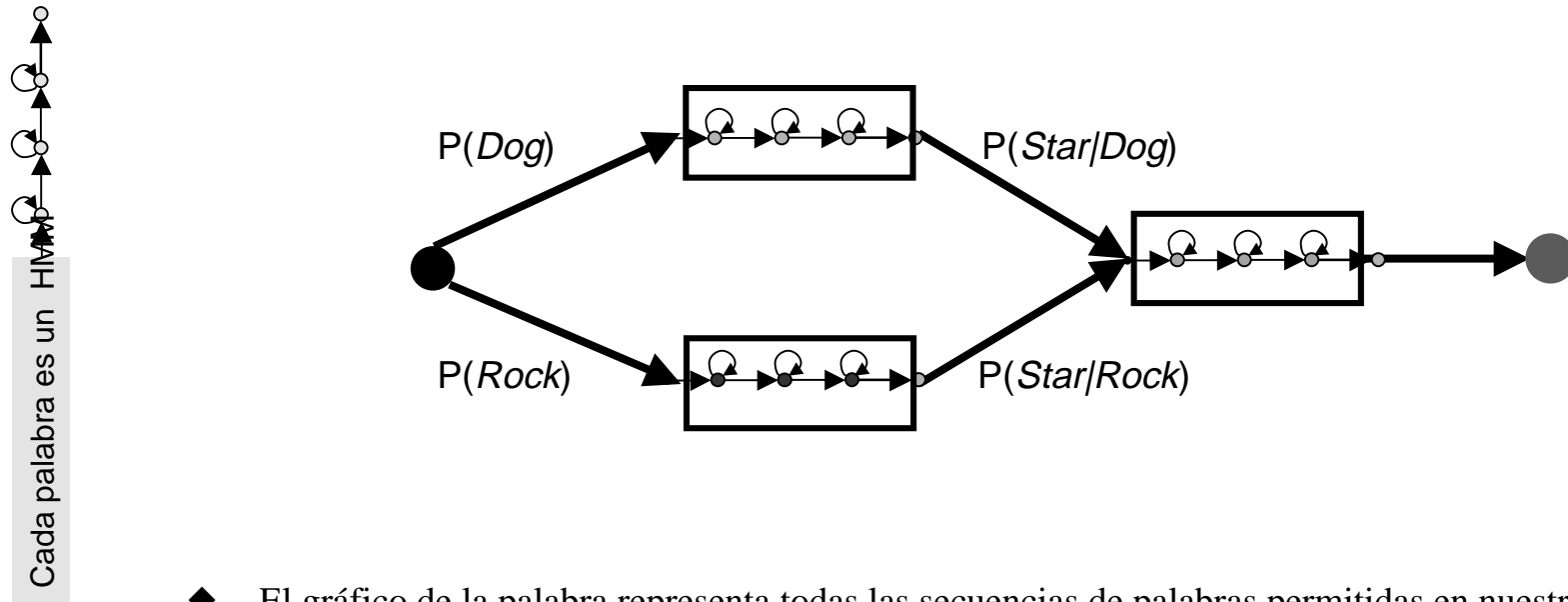
- No surgen incoherencias si las puntuaciones de ruta cambian monótonicamente con una longitud de ruta creciente
- Esto puede garantizarse normalizando todos los valores de densidad de salida de estado para un vector en tiempo t con relación a la más alta densidad valorada en t

Diseño de estructuras gráficas óptimas para el lenguaje HMM

Lenguaje HMM para secuencias de palabras de longitud fija



Lenguaje HMM para secuencias de palabras de longitud fija

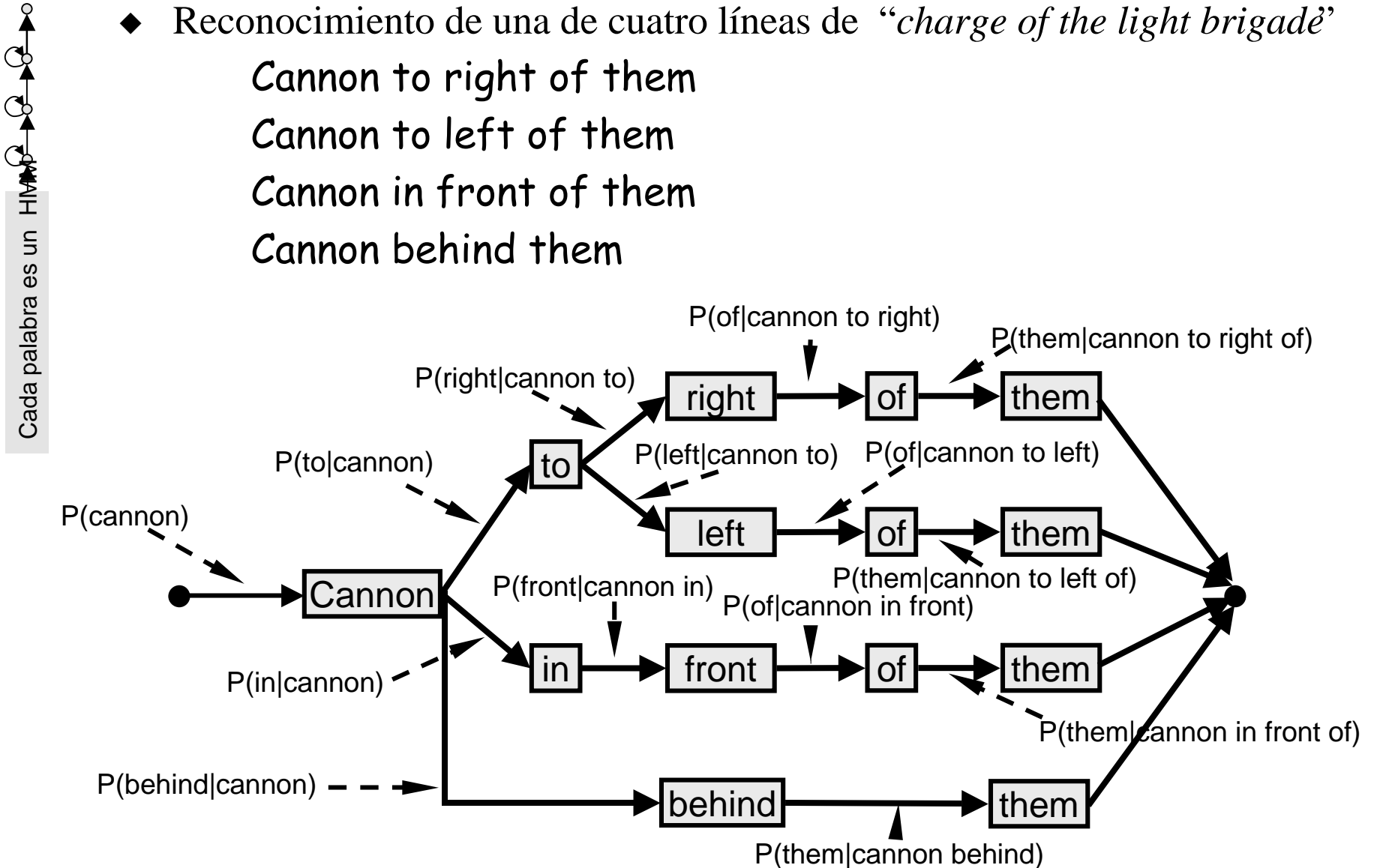


- ◆ El gráfico de la palabra representa todas las secuencias de palabras permitidas en nuestro ejemplo
 - El conjunto de todas las secuencias de palabras permitidas representa el “lenguaje ” permitido
- ◆ En un nivel más detallado, el dibujo representa un HMM compuesto por los HMM para todas las palabras en el gráfico de la palabra
 - Este es el “Lenguaje HMM” – el HMM para el lenguaje permitido completo
- ◆ El lenguaje HMM representa el eje vertical del trellis
 - Es el trellis, y NO el lenguaje HMM, el que se busca para la mejor ruta

Lenguajes HMM para secuencias de palabras de longitud fija

- ◆ Reconocimiento de una de cuatro líneas de “*charge of the light brigadè*”

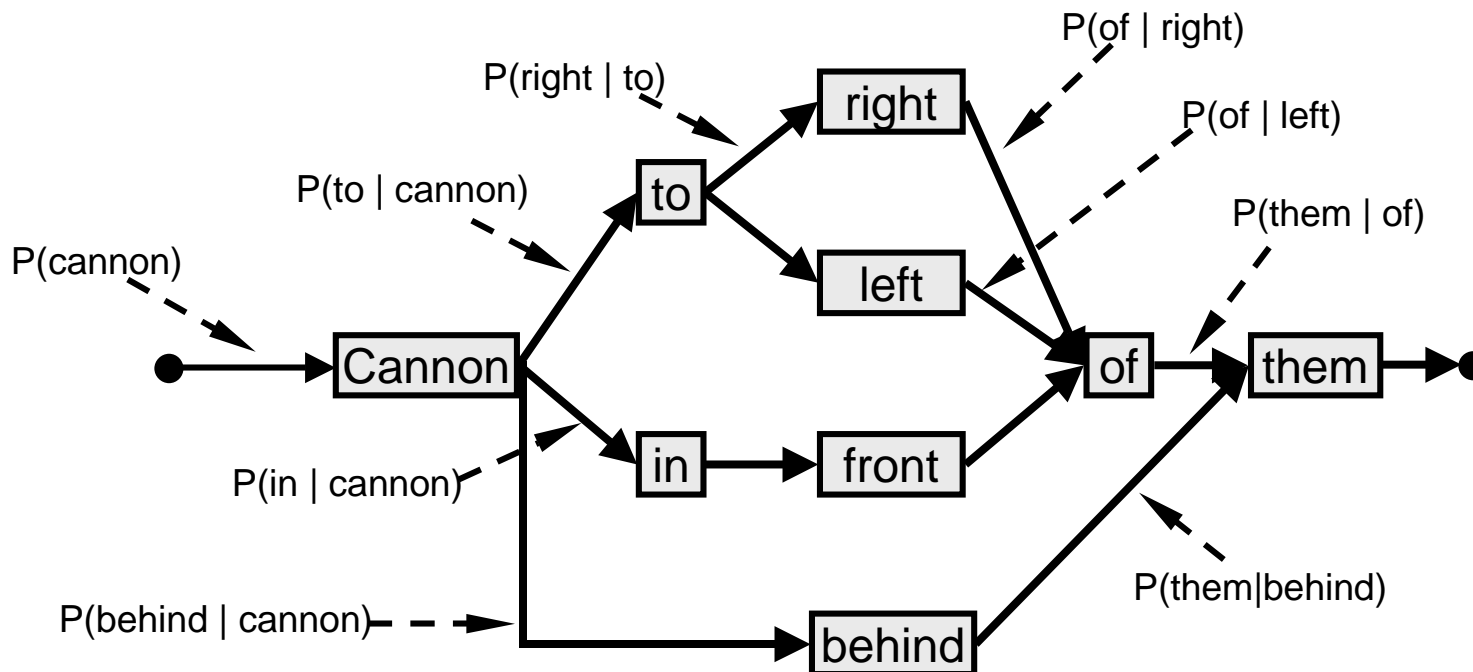
Cannon to right of them
 Cannon to left of them
 Cannon in front of them
 Cannon behind them



Simplificación del lenguaje HMM mediante modelos de lenguaje de contexto inferior

Cada palabra es un HMM

- ◆ Reconocimiento de una de cuatro líneas de “*charge of the light brigadè*”
- ◆ Si la probabilidad de una palabra sólo depende de la palabra anterior, el gráfico se puede plegar:
 - ej., $P(\text{them} \mid \text{cannon to right of}) = P(\text{them} \mid \text{cannon to left of}) = P(\text{cannon} \mid \text{of})$



Problema de reconocimiento refactorizado para lenguajes HMM que se amalgaman con modelos de lenguaje de contexto inferior

- ◆ Nuestro nuevo enunciado del problema con una necesidad inferior del modelo de lenguaje:

$word1, word2, \dots =$

$$\operatorname{argmax}_{wd_1, wd_2, \dots} \{ \operatorname{argmax}_{s_1, s_2, \dots, s_T} P(wd_1, wd_2, \dots) P(X_1, X_2, \dots, X_T, s_1, s_2, \dots, s_T / wd_1, wd_2, \dots) \}$$

- ◆ El término de probabilidad puede factorizarse en palabras individuales como

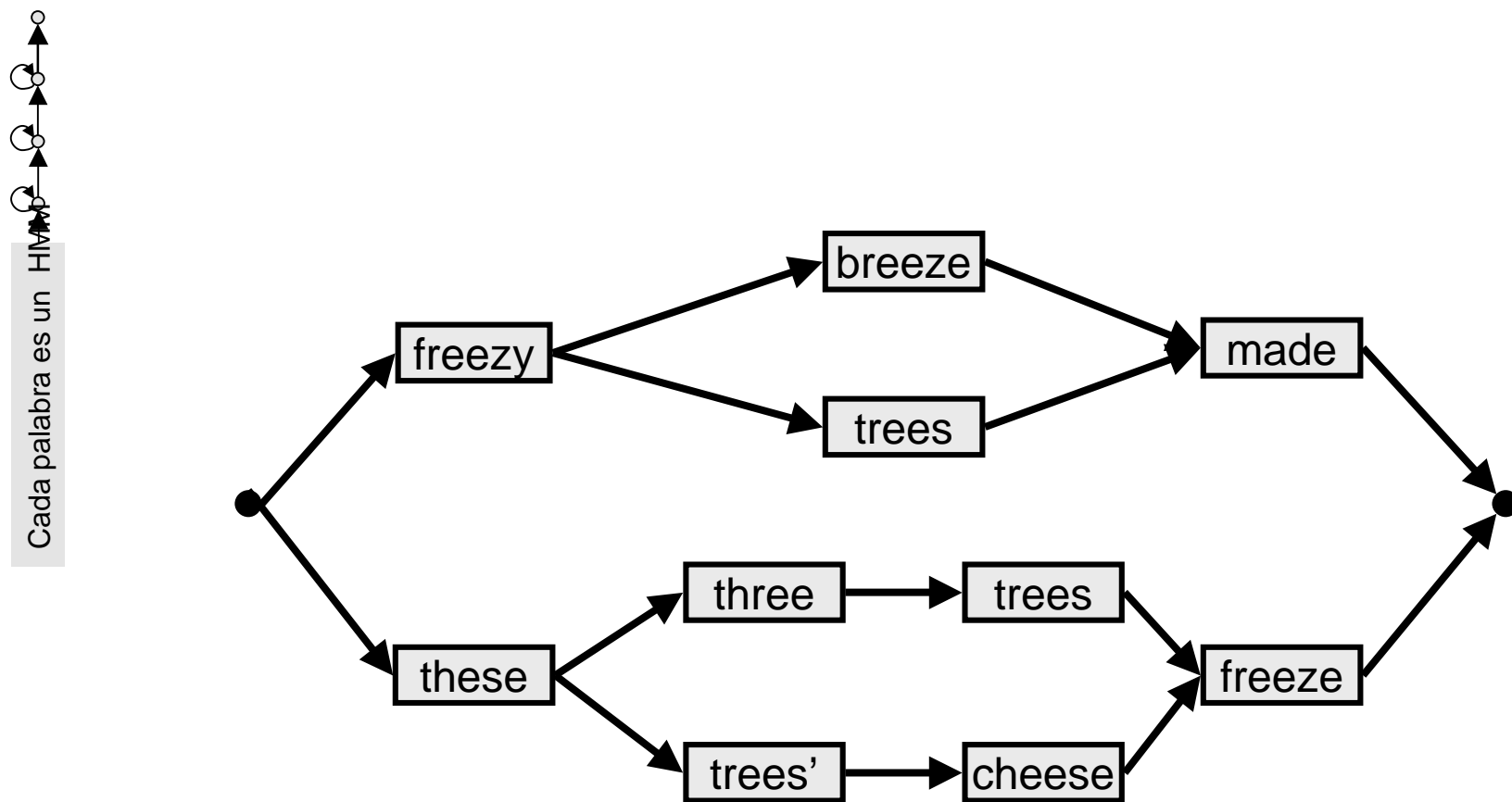
$$P(wd_1) P(X_{wd1}, S_{wd1} / wd_1) \cdot P(wd_2 / wd_1) P(X_{wd2}, S_{wd2} / wd_2) \cdot \\ P(wd_3 / wd_1, wd_2) P(X_{wd3}, S_{wd3} / wd_3) \dots P(wd_N / wd_1 \dots wd_{N-1}) P(X_{wdN}, S_{wdN} / wd_N)$$

- ◆ Suponer contextos más allá de una determinada longitud K no tiene importancia

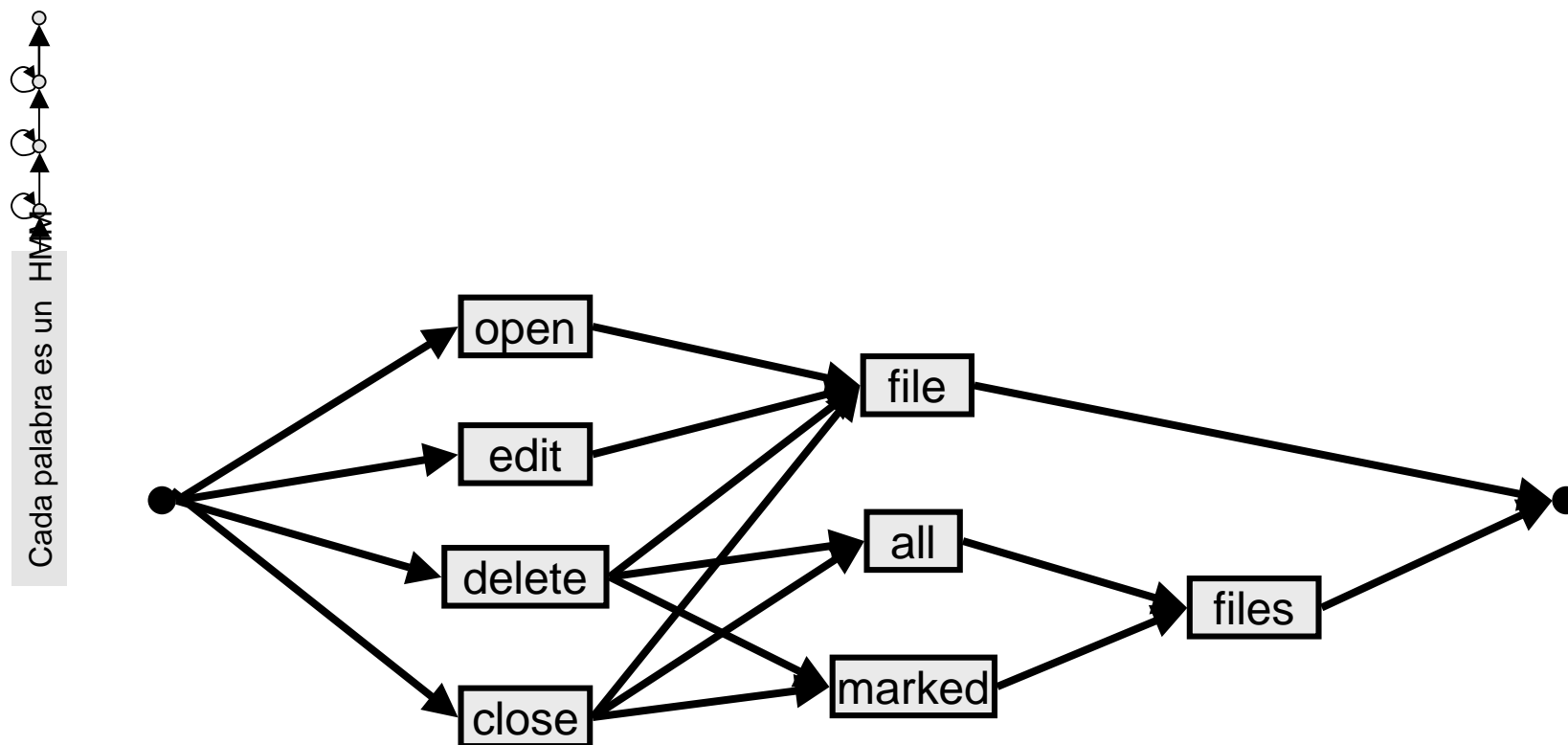
$$P(wd_N | wd_{N-1}, wd_{N-2}, \dots, wd_{N-K}, \dots, wd_1) = P(wd_N | wd_{N-1}, wd_{N-2}, \dots, wd_{N-K})$$

- ◆ Los nodos con la misma historia de palabras $wd_{N-1}, wd_{N-2}, \dots, wd_{N-K}$ pueden plegarse

Lenguajes HMM para secuencias de palabras de longitud fija: basado en una gramática para el Dr. Seuss



Lenguajes HMM para secuencias de palabra de longitud fija: gramática de comandos y de control



Lenguajes HMM para secuencias de palabras arbitrariamente largas

- ◆ Los lenguajes anteriores eligen entre un conjunto finito de secuencias de palabras conocidas
- ◆ Las secuencias de palabras pueden ser de longitud arbitraria
 - Ej. conjunto de todas las secuencias de palabras que constan de un número arbitrario de repeticiones de la palabra *bang* (estrépito)

bang

bang bang

bang bang bang

bang bang bang bang

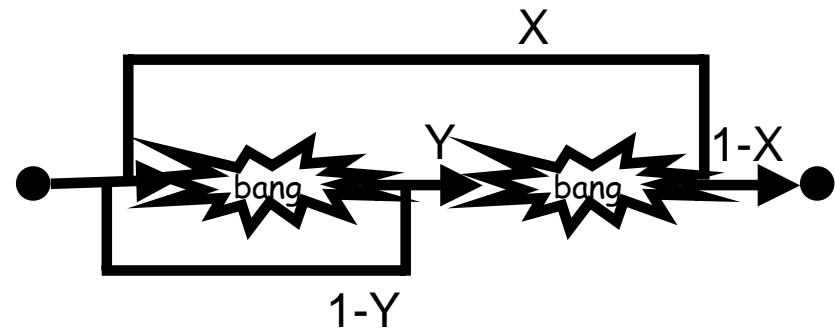
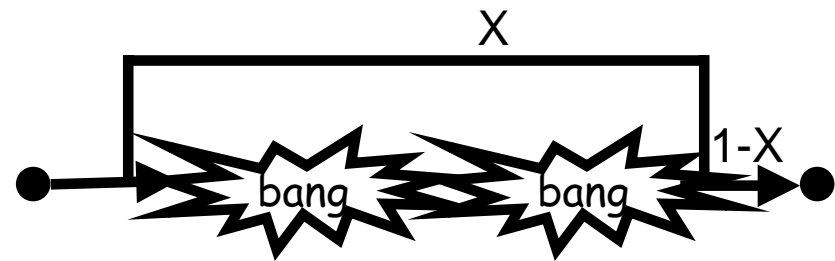
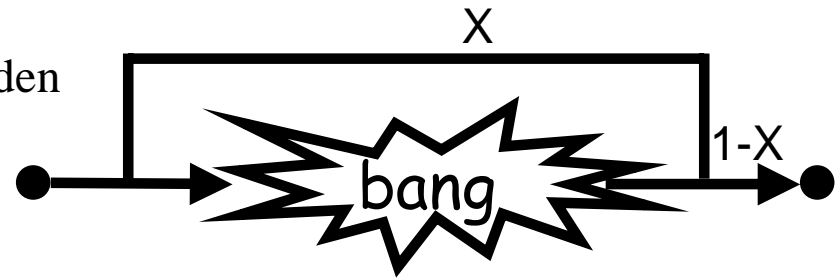
.....

- La formación de gráficos de secuencias de palabras explícitas del tipo que hasta ahora hemos visto, no es posible
 - ▶ El número de secuencias posibles (con probabilidad *a-priori* no nula) es potencialmente infinito
 - ▶ Incluso si la longitud de la secuencia más larga está restringida, el gráfico continuará siendo grande

Lenguajes HMM para secuencias de palabras arbitrariamente largas

Cada palabra es un HMM

- ◆ Las secuencias de palabras arbitrarias pueden modelarse con bucles bajo ciertas suposiciones. Ej.:
- ◆ Un “*bang*” puede ir seguido de otro “*bang*” con probabilidad $P(\text{“bang”})$.
 - $P(\text{“bang”}) = X$; $P(\text{Termination}) = 1-X$;
- ◆ Los *bangs* pueden darse sólo en pares con probabilidad X
- ◆ Un gráfico más complejo permite más patrones complicados
- ◆ Usted puede extender esta lógica a otros vocabularios donde el hablante diga más cosas además de “*bang*”
 - ej. “*bang bang you’re dead*” (Bang bang, ¡muerto!)



Lenguaje HMM para secuencias de palabras arbitrariamente largas

- ◆ Los grupos limitados de secuencias de palabras con vocabulario restringido son realísticos
 - Típicamente en situaciones de comando y control
 - Ejemplo: manejando un TV remoto
 - Sistemas de diálogos simples
 - Cuando el conjunto de respuestas permitidas a una pregunta es restringido
- ◆ Secuencias de palabras de longitud no restringida: LENGUAJE NATURAL
 - Decodificadores de última generación de enormes vocabularios

Lenguajes HMM para el lenguaje natural: con representaciones de unigrama

- ◆ Una cantidad de modelos de palabras:

- Cualquier palabra es posible después de cualquier otra
- La probabilidad de una palabra es independiente de las palabras que la precedan o sucedan
- Conocido también como modelo UNIGRAMA

$$P(\textit{Star}) = P(\textit{Star} \mid \textit{Dog}) = P(\textit{Star} \mid \textit{Rock}) = P(\textit{Star} \mid \textit{When you wish upon a})$$

$$P(\textit{When you wish upon a star}) =$$

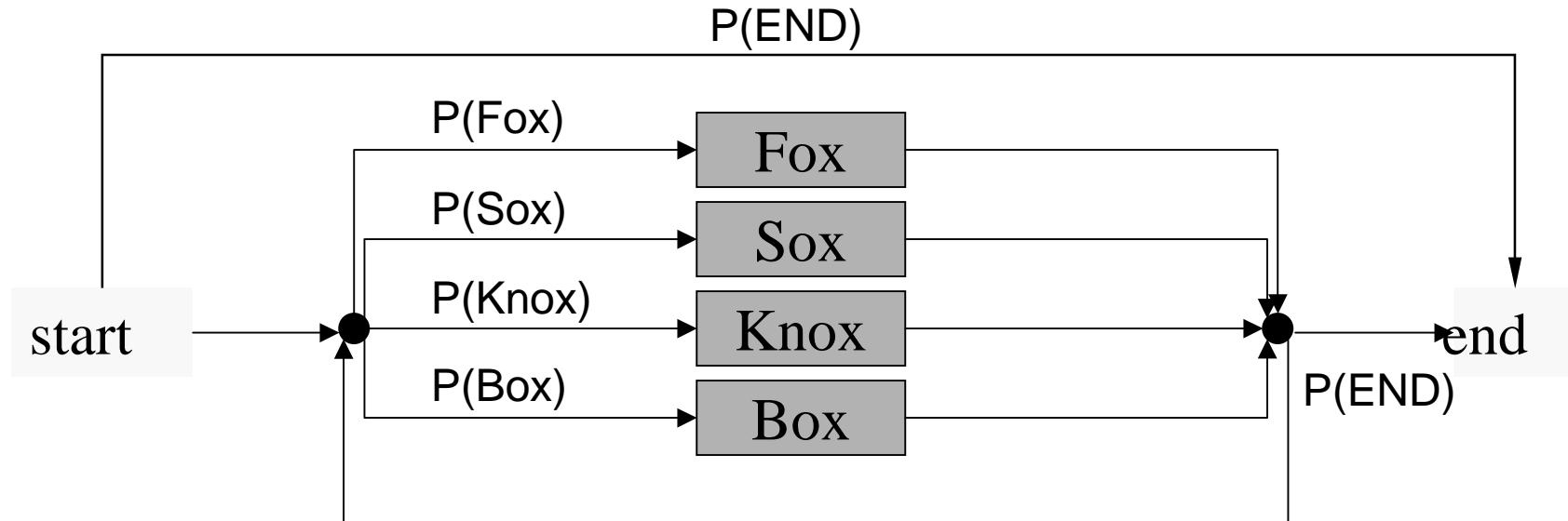
$$P(\textit{When}) P(\textit{you}) P(\textit{wish}) P(\textit{upon}) P(\textit{a}) P(\textit{star}) P(\textit{END})$$

- ◆ “END” es un símbolo especial que indica el fin de la secuencia de palabras
- ◆ P(END) es necesario – sin él, la secuencia de palabras nunca acabaría
 - La probabilidad total de todas las secuencias de palabras posibles debe sumar 1.0
 - Sólo si P(END) se define explícitamente con la probabilidad total = 1.0

Lenguajes HMM para el lenguaje natural: ejemplo de un gráfico que emplea representaciones de unigrama

Cada palabra es un HMM

- ◆ Vocabulario: “fox”, “sox”, “knox”, “box”
- ◆ Secuencias de longitud arbitraria con disposiciones arbitrarias de estas cuatro palabras
- ◆ Probabilidades reales:
 - $P(\text{fox}), P(\text{sox}), P(\text{knox}), P(\text{box}), P(\text{END})$
 - $P(\text{fox}) + P(\text{sox}) + P(\text{knox}) + P(\text{box}) + P(\text{END}) = 1.0$



• Los puntos blancos son estados “no emitenes” que no están asociados con observaciones

Lenguajes HMM para el lenguaje natural: representaciones de bigrama

- ◆ Un modelo unigrama sólo resulta útil cuando no se asume una dependencia estadística entre palabras adyacentes
 - O, alternativamente, cuando los datos empleados para aprender estas dependencias son demasiado pequeños para que resulten fiables
 - Aprendizaje de dependencias de palabra: Más adelante en el programa
- ◆ En el lenguaje natural, la probabilidad de aparición de una palabra depende de las palabras anteriores
- ◆ Modelo de lenguaje bigrama: la probabilidad de una palabra depende de la palabra anterior
- ◆ $P(\text{Star} \mid \text{A Rock}) = P(\text{Star} \mid \text{The Rock}) = P(\text{Star} \mid \text{Rock})$
- ◆ No se precisa que $P(\text{Star} \mid \text{Rock})$ sea igual a $P(\text{Star} \mid \text{Dog})$
 - De hecho, se supone que ambos términos no están relacionados

Lenguajes HMM para el lenguaje natural: representaciones de bigrama

- ◆ Ejemplo de bigrama simple:
 - Vocabulario: START, “odd”, “even”, END
 - $P(\text{odd} \mid \text{START}) = a$, $P(\text{even} \mid \text{START}) = b$, $P(\text{END} \mid \text{START}) = c$
 - $a+b+c = 1.0$
 - $P(\text{odd} \mid \text{even}) = d$, $P(\text{even} \mid \text{even}) = e$, $P(\text{END} \mid \text{even}) = f$
 - $d+e+f = 1.0$
 - $P(\text{odd} \mid \text{odd}) = g$, $P(\text{even} \mid \text{odd}) = h$, $P(\text{END} \mid \text{odd}) = i$
 - $g+h+i = 1.0$

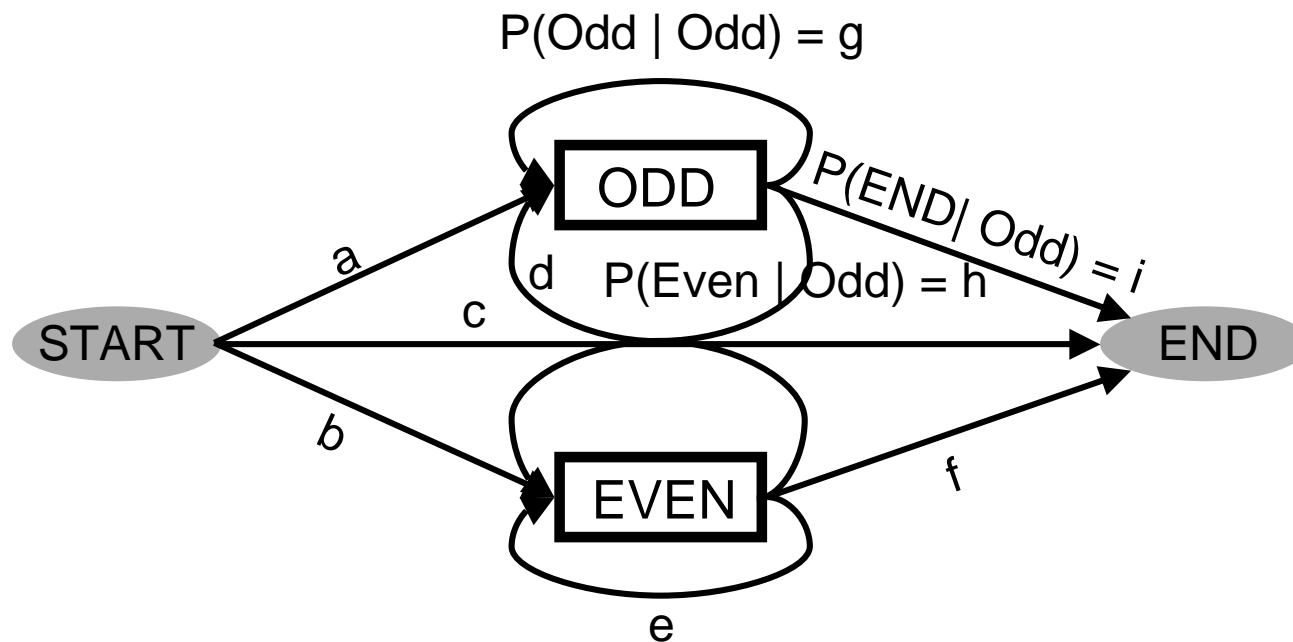
- ◆ START es un símbolo especial que indica el comienzo del enunciado
 - $P(\text{word} \mid \text{START})$ es la probabilidad de que el enunciado comience con la palabra
 - $\text{Prob}(\text{“odd even even odd”}) =$
 $P(\text{odd} \mid \text{START}) P(\text{even} \mid \text{odd}) P(\text{even} \mid \text{even}) P(\text{odd} \mid \text{even}) P(\text{END} \mid \text{odd})$

- ◆ Se puede demostrar que la probabilidad total de todas las secuencias de palabras de todas las longitudes es 1.0
 - Nuevamente, la definición de los símbolos START y END, y de todos los bigramas que impliquen a los dos, es crucial

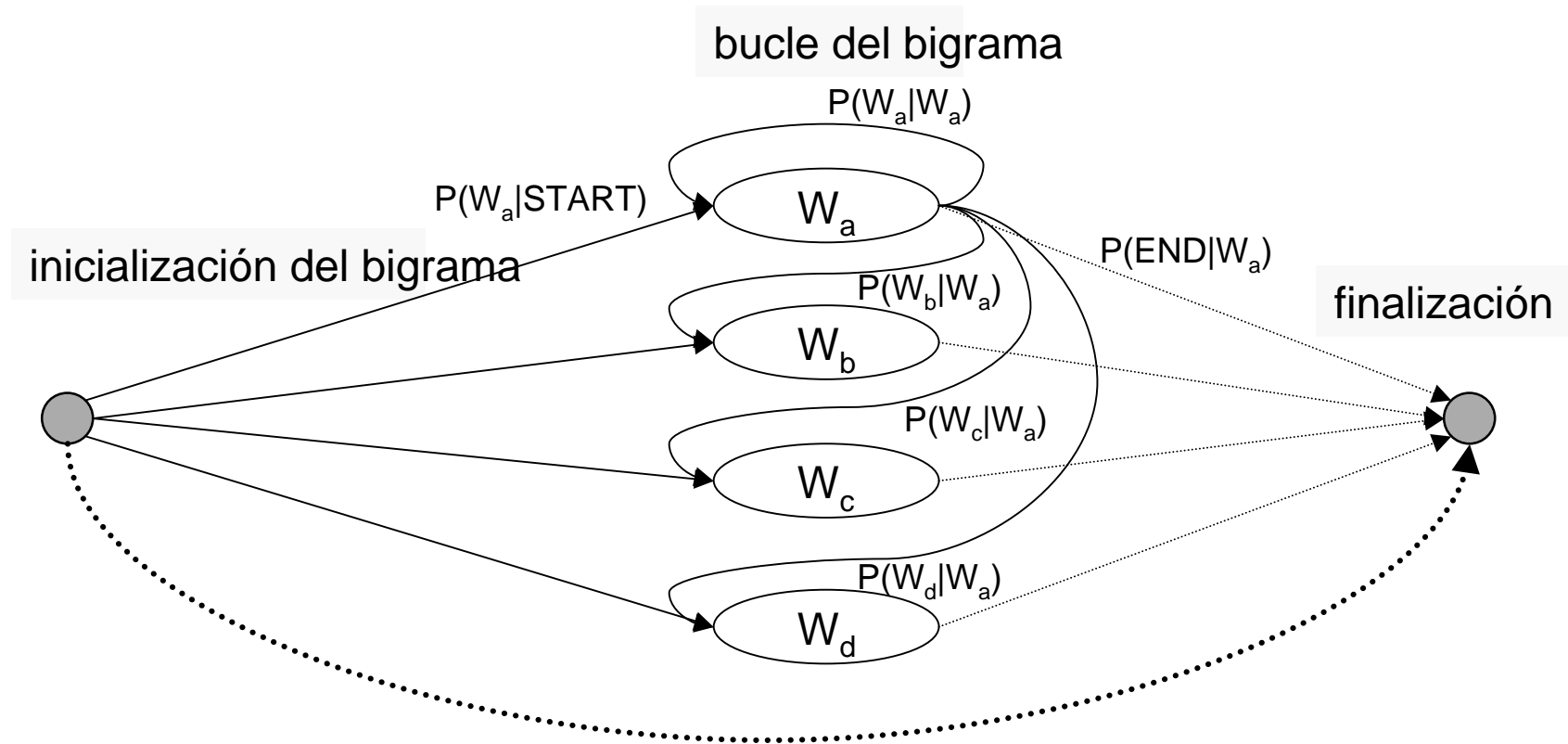
Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de bigramas

Cada palabra es un HMM

- ◆ Las flechas desde “START” contienen probabilidades de palabra dependientes de START
- ◆ Las flechas desde “Even” contienen probabilidades de palabra dependientes de “Even”
- ◆ Las flechas desde “Odd” contienen probabilidades de palabra dependientes de “Odd”



Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de bigramas



Lenguajes HMM para el lenguaje natural: representaciones de trigramas

- ◆ Asunción: La probabilidad de una palabra depende de las dos palabras precedentes

$$P(\text{waltzing} \mid \text{you'll come a}) = P(\text{waltzing} \mid \text{who'll come a}) = P(\text{waltzing} \mid \text{come a})$$

$$\begin{aligned} P(\text{you'll come a waltzing matilda with me}) = & \\ & P(\text{you'll} \mid \text{START}) * P(\text{come} \mid \text{START you'll}) * \\ & P(\text{a} \mid \text{you'll come}) * P(\text{waltzing} \mid \text{come a}) * \\ & P(\text{matilda} \mid \text{a waltzing}) * P(\text{with} \mid \text{waltzing matilda}) * \\ & P(\text{me} \mid \text{matilda with}) * P(\text{END} \mid \text{with me}) \end{aligned}$$

- ◆ Para cualquier secuencia de palabras $w_1, w_2, w_3, w_4, w_5 \dots w_N$

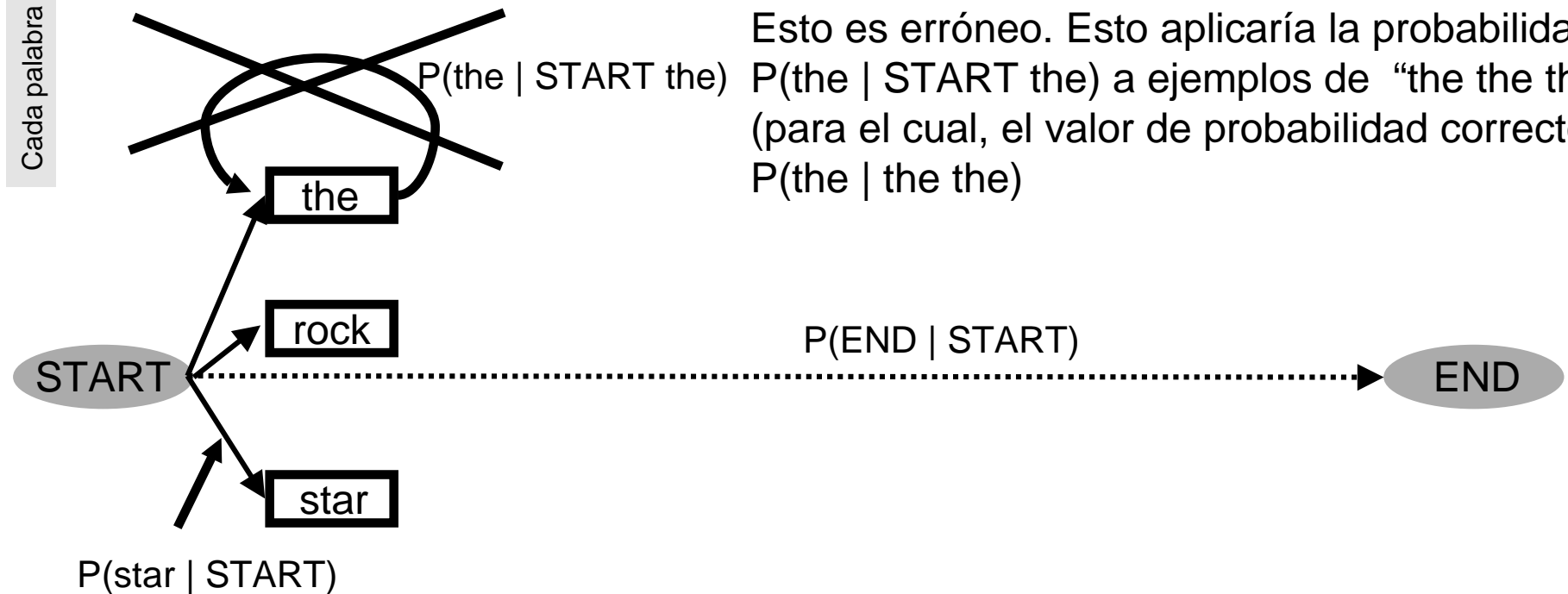
- ◆
$$P(w_1, w_2 \dots w_N) = P(w_1 \mid \text{START}) P(w_2 \mid \text{START } w_1) P(w_3 \mid w_1 w_2) \dots P(\text{END} \mid w_{N-1} w_N)$$

- Observe que el PRIMER término $P(\psi \mid \text{START})$ es un término *bigrama*. Todo lo demás son trigramas.

Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas

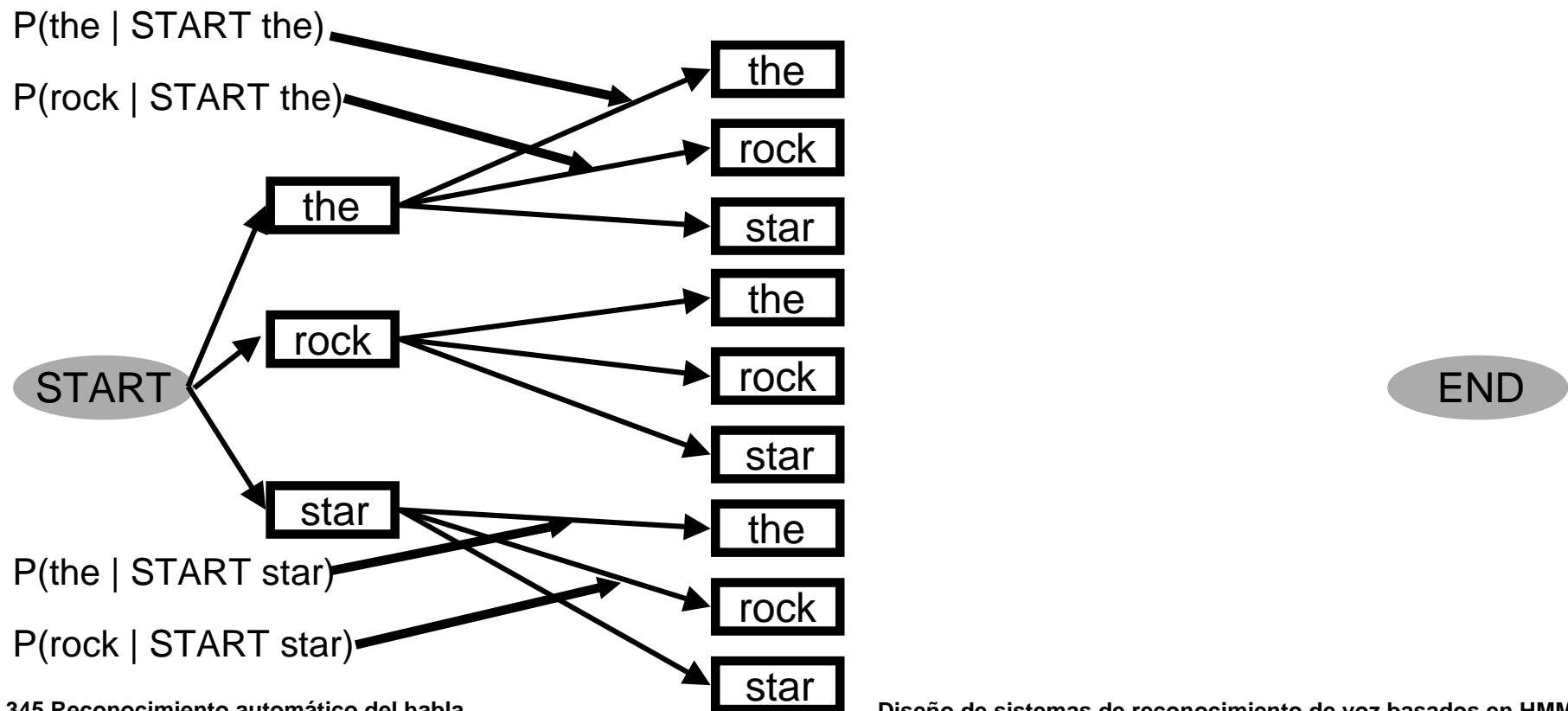
- ◆ Explicación con ejemplo: vocabulario de tres palabras “the”, “rock”, “star”
 - El gráfico comienza inicialmente con bigramas de START, dado que nada precede a START
 - Trigramas de “START the”..

Cada palabra es un HMM



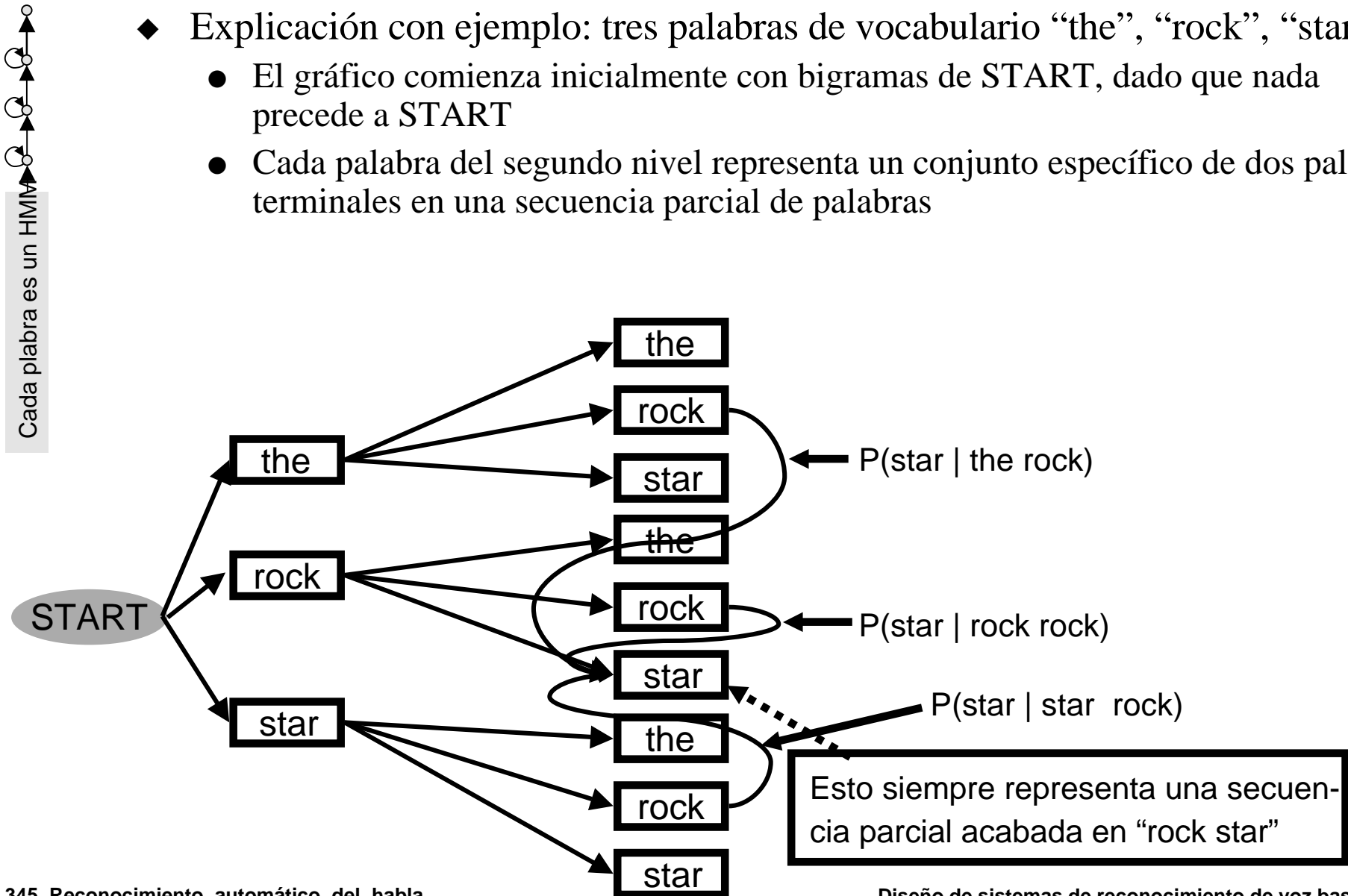
Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas

- ◆ Explicación con ejemplo: tres palabras del vocabulario “the”, “rock”, “star”
 - El gráfico comienza inicialmente con bigramas de START, dado que nada precede a START
 - Los trigramas para las secuencias “START word”
 - Es necesario un nuevo ejemplo de cada palabra para asegurar que los dos símbolos precedentes son “START word”



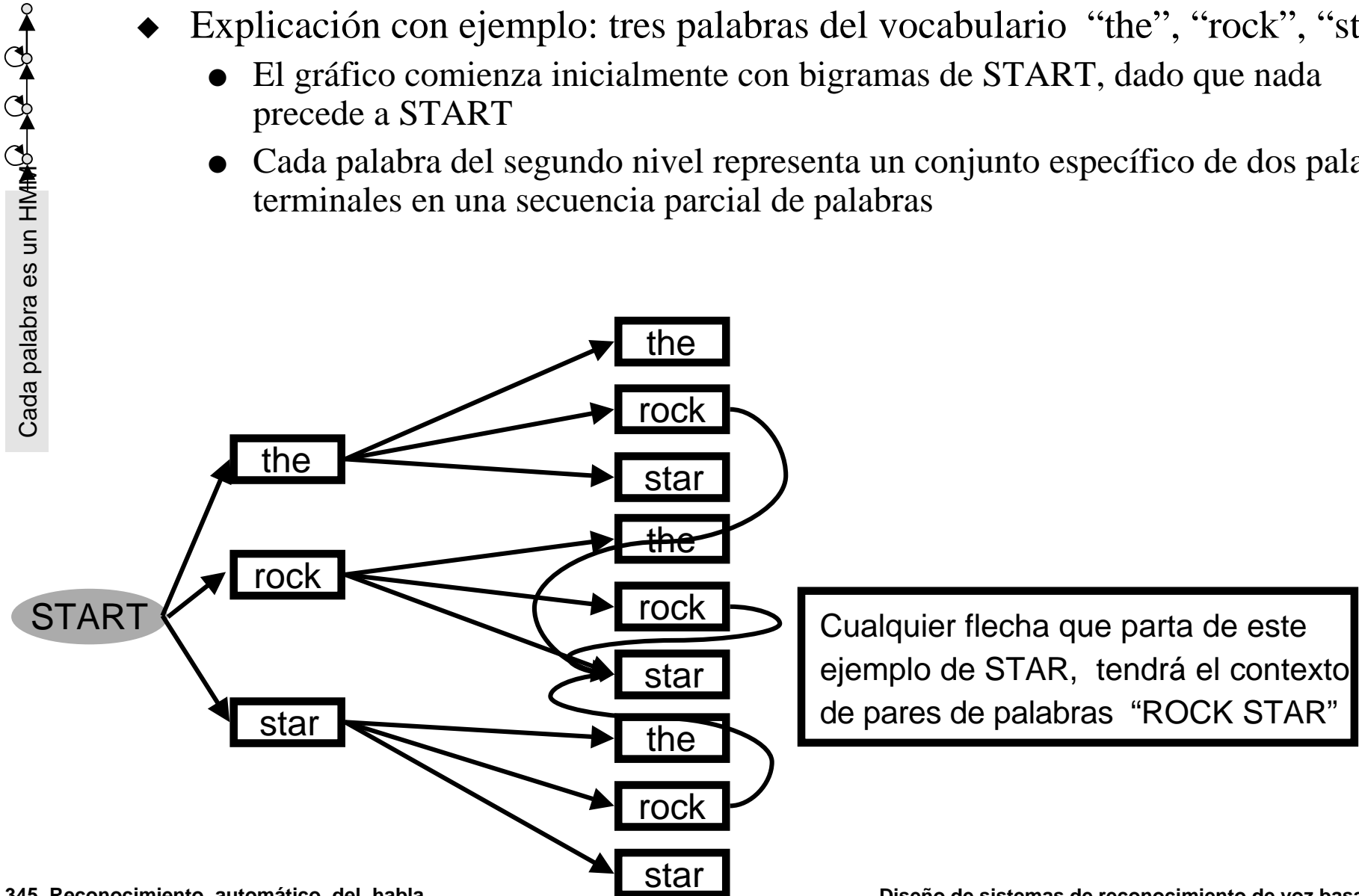
Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas

- ◆ Explicación con ejemplo: tres palabras de vocabulario “the”, “rock”, “star”
 - El gráfico comienza inicialmente con bigramas de START, dado que nada precede a START
 - Cada palabra del segundo nivel representa un conjunto específico de dos palabras terminales en una secuencia parcial de palabras



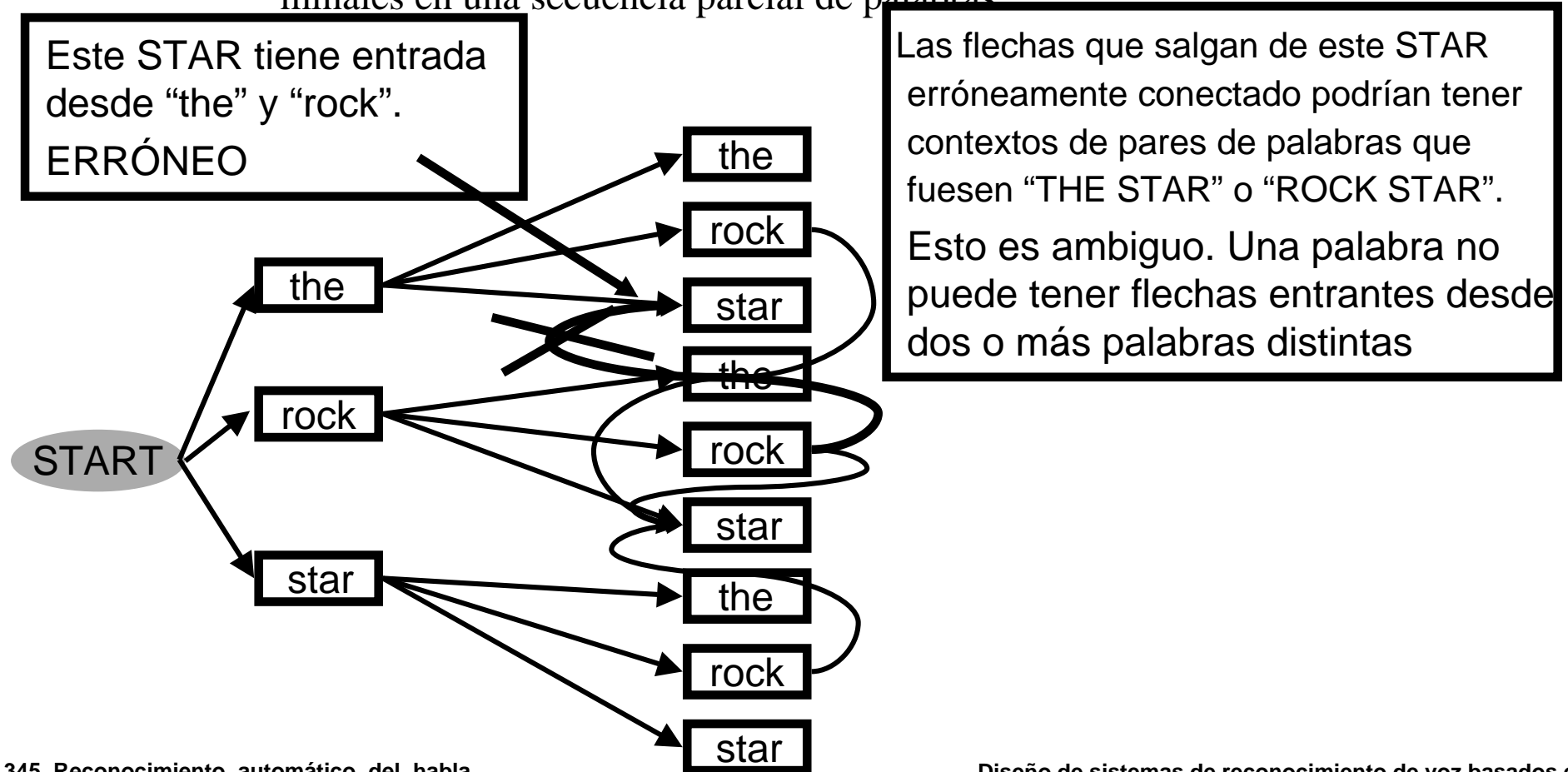
Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas

- ◆ Explicación con ejemplo: tres palabras del vocabulario “the”, “rock”, “star”
 - El gráfico comienza inicialmente con bigramas de START, dado que nada precede a START
 - Cada palabra del segundo nivel representa un conjunto específico de dos palabras terminales en una secuencia parcial de palabras

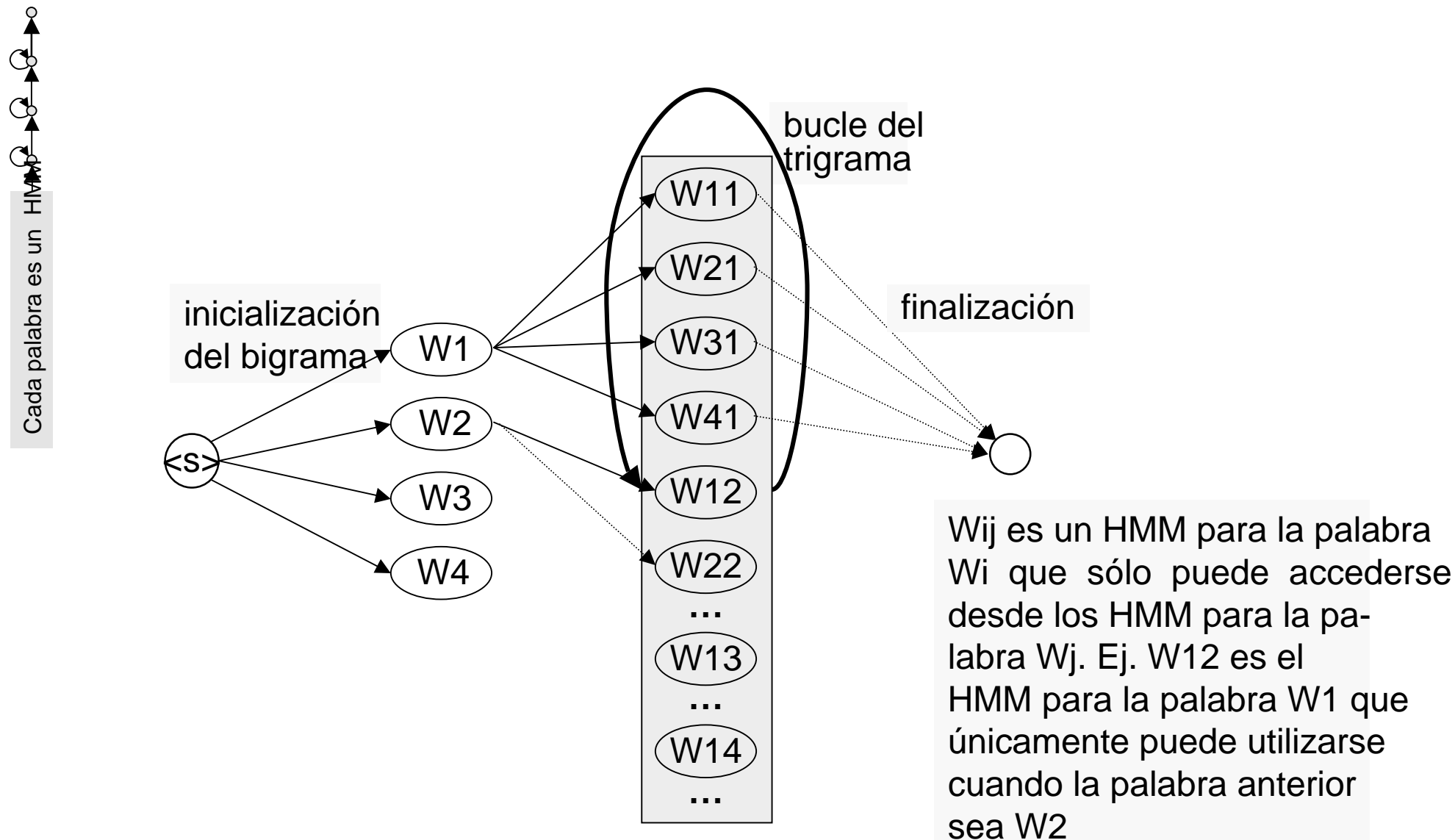


Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas

- ◆ Explicación con ejemplo: tres palabras del vocabulario “the”, “rock”, “star”
 - El gráfico comienza inicialmente con bigramas de START, dado que nada precede a START
 - Cada palabra del segundo nivel representa un conjunto específico de dos palabras terminales en una secuencia parcial de palabras



Lenguajes HMM para el lenguaje natural: construcción de gráficos para incorporar representaciones de trigramas



Lenguajes HMM para el lenguaje natural: representaciones genéricas de n-grama

- ◆ La lógica puede extenderse
- ◆ Una estructura descodificadora de trigramas para un vocabulario de D palabras, requiere D ejemplos de palabras en el primer nivel y D^2 ejemplos de palabras en el segundo nivel
 - Deben instanciarse un total de $D(D+1)$ modelos de palabra
 - También cabe la posibilidad de otras estructuras más costosas
- ◆ Una estructura descodificadora de N-grama requerirá:
 - $D + D^2 + D^3 \dots D^{N-1}$ ejemplos de palabras
 - Los arcos deben incorporarse de modo que la salida desde un ejemplo de la palabra en el nivel $(N-1)$ siempre represente una secuencia de palabras con la misma secuencia de salida de $N-1$ palabras

Próxima clase

- ◆ Cómo el uso de unidades de subpalabras complica la estructura de los lenguajes HMM en los decodificadores