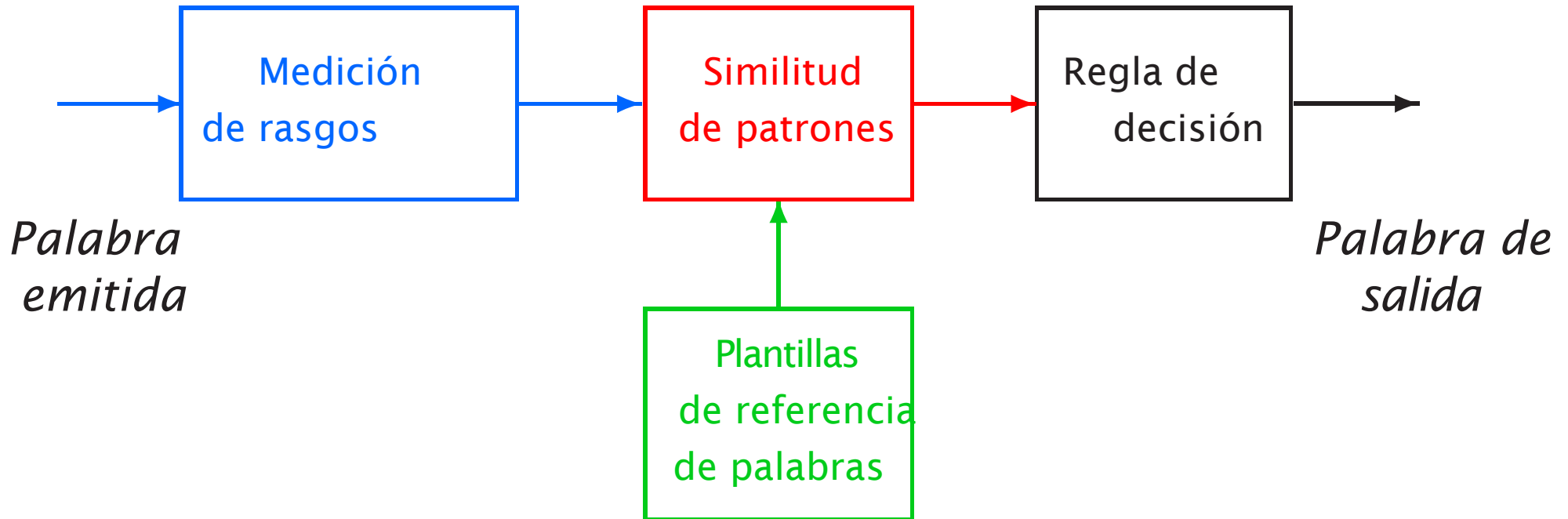


# Distorsión dinámica temporal y búsqueda

- Distorsión dinámica temporal
- Búsqueda
  - Algoritmos de búsqueda gráfica
  - Algoritmos de programación dinámicos

# Coincidencia de plantilla basada en palabras



- Representación de la palabra completa:
  - No existe un concepto explícito de unidades de subpalabra (ej., fonos).
  - Las palabras entre sí no comparten nada.
- Se utiliza en el reconocimiento de palabras aisladas y en el del discurso hablado.
- Tuvo éxito desde finales de la década de los 70 hasta mediados de los 80.

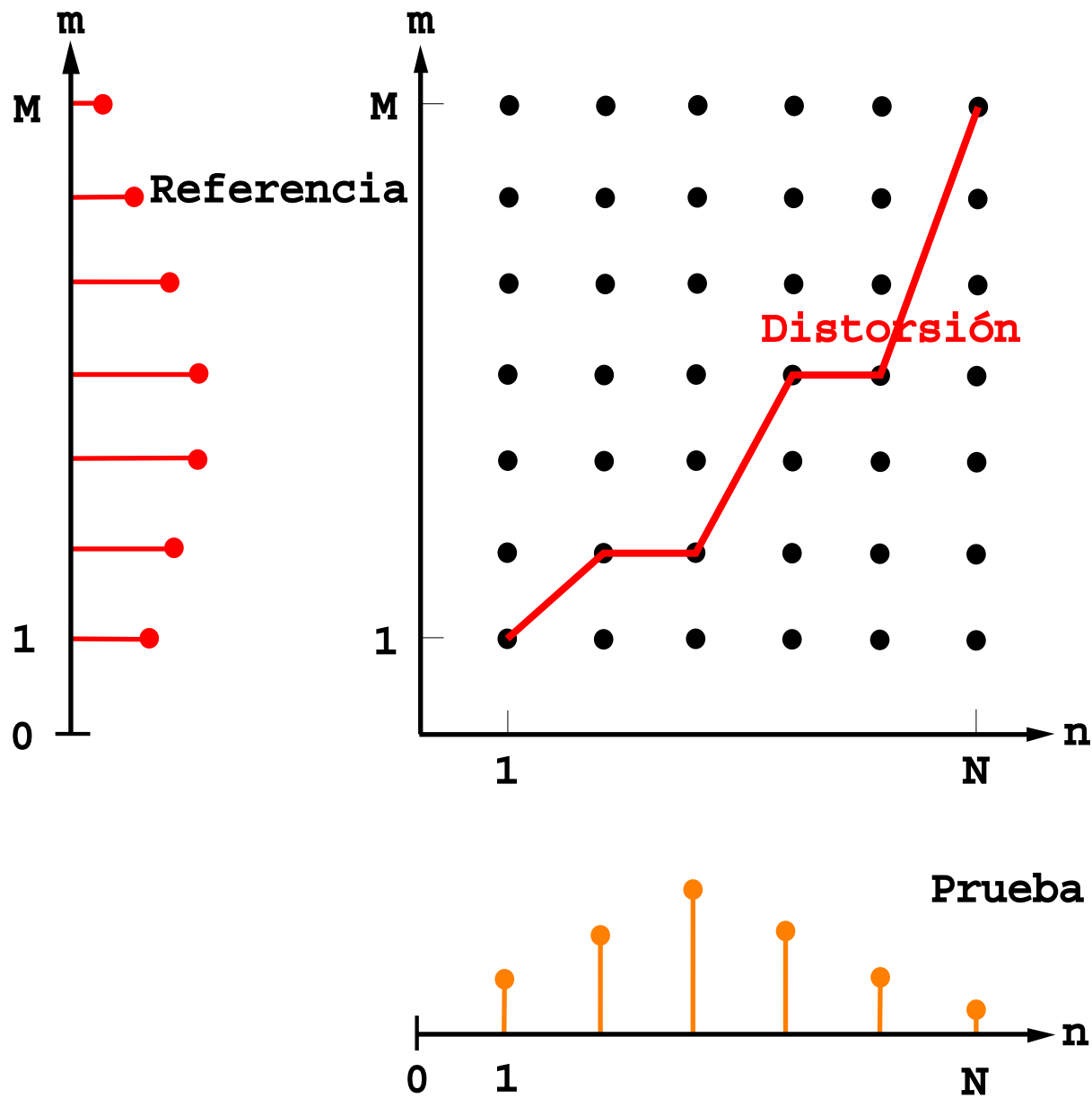
# Mecanismo de coincidencia o encaje de plantillas

- El patrón de prueba  $\mathbf{T}$  y los patrones de referencia  $\{\mathbf{R}_1, \dots, \mathbf{R}_V\}$ , están representados por secuencias de mediciones de rasgos.
- La similitud de patrones está determinada por el **alineamiento** del patrón de prueba  $\mathbf{T}$ , con el patrón de referencia  $\mathbf{R}_v$ , con distorsión  $\mathcal{D}(\mathbf{T}, \mathbf{R}_v)$
- La regla de decisión selecciona el patrón de referencia  $\mathbf{R}^*$ , con la menor distorsión de alineamiento  $\mathcal{D}(\mathbf{T}, \mathbf{R}^*)$

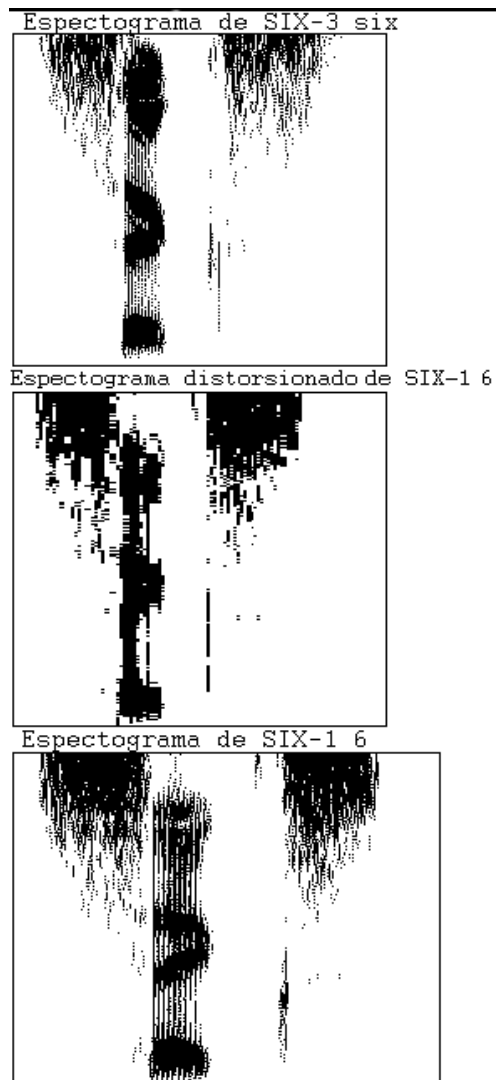
$$\mathbf{R}^* = \arg \min_v \mathcal{D}(\mathbf{T}, \mathbf{R}_v)$$

- La técnica de la **distorsión dinámica temporal** (DTW) se utiliza para calcular la mejor distorsión de alineamiento posible  $\phi_v$ , entre  $\mathbf{T}$  y  $\mathbf{R}_v$ , junto con la distorsión asociada  $\mathcal{D}(\mathbf{T}, \mathbf{R}_v)$

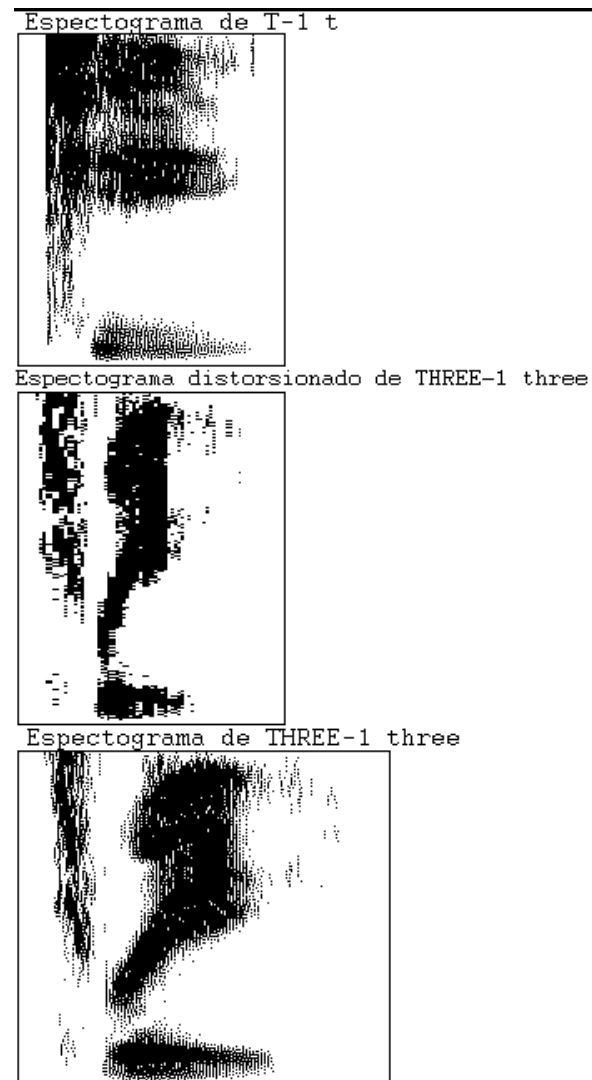
# Ejemplo de alineamiento



# Ejemplos de alineamiento digital



Encaje



Desencaje

## Distorsión dinámica temporal (DTW)

- **Objetivo:** un alineamiento óptimo entre las secuencias de longitud variable  $\mathbf{T} = \{\mathbf{t}_1, \dots, \mathbf{t}_N\}$  y  $\mathbf{R} = \{\mathbf{r}_1, \dots, \mathbf{r}_M\}$
- La distorsión total  $\mathcal{D}(\mathbf{T}, \mathbf{R})$  se basa en una suma de distancias locales entre elementos  $d(\mathbf{t}_i, \mathbf{r}_j)$
- Una distorsión de aleamiento especial  $\phi$ , alinea  $\mathbf{T}$  y  $\mathbf{R}$  mediante un mapeo de punto a punto  $\phi = (\phi_t, \phi_r)$ , de longitud  $K_\phi$

$$\mathbf{t}_{\phi_t(k)} \iff \mathbf{r}_{\phi_r(k)} \quad 1 \leq k \leq K_\phi$$

- El alineamiento óptimo minimiza la distorsión global.

$$\mathcal{D}(\mathbf{T}, \mathbf{R}) = \min_{\phi} \mathcal{D}_{\phi}(\mathbf{T}, \mathbf{R})$$

$$\mathcal{D}_{\phi}(\mathbf{T}, \mathbf{R}) = \frac{1}{M_{\phi}} \sum_{k=1}^{K_{\phi}} d(\mathbf{t}_{\phi_t(k)}, \mathbf{r}_{\phi_r(k)}) m_k$$

# Cuestiones sobre las técnicas de DTW

- Restricciones de punto final:

$$\phi_t(1) = \phi_r(1) = 1 \quad \phi_t(K) = N \quad \phi_r(K) = M$$

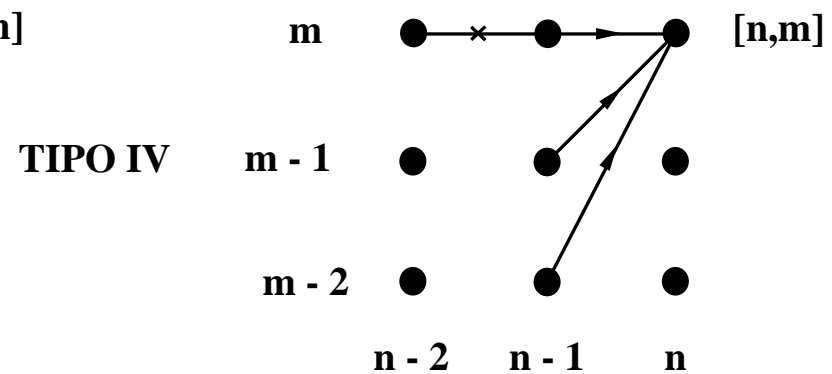
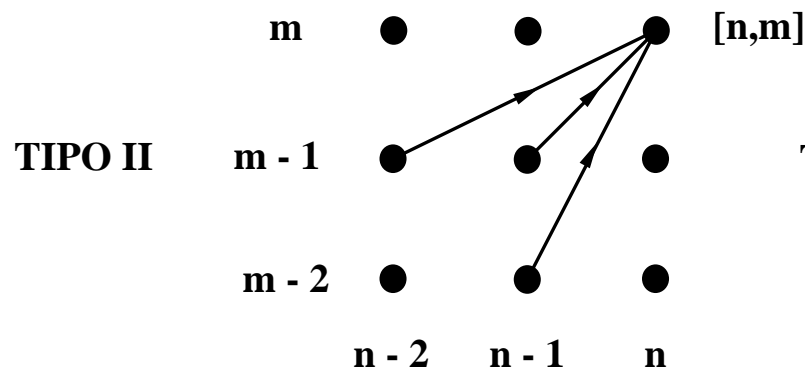
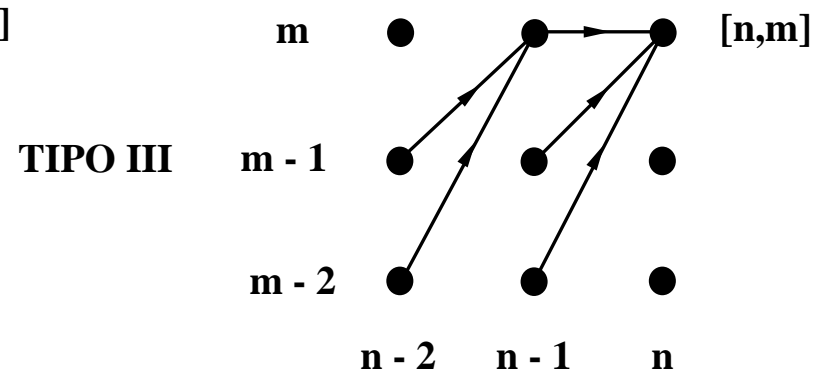
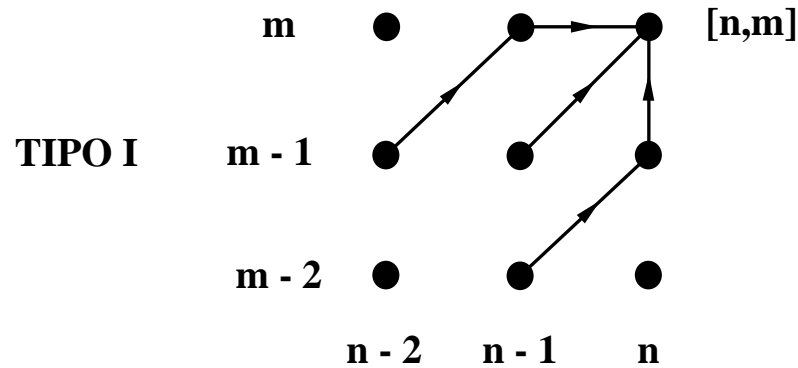
- Monotonicidad:

$$\phi_t(k+1) \geq \phi_t(k) \quad \phi_r(k+1) \geq \phi_r(k)$$

- Los pesos de la trayectoria  $m_k$ , pueden influenciar la forma de la trayectoria óptima.
- El factor de normalización de trayectoria  $M_\phi$ , permite la comparación entre distintas distorsiones (ej., con diferentes longitudes).

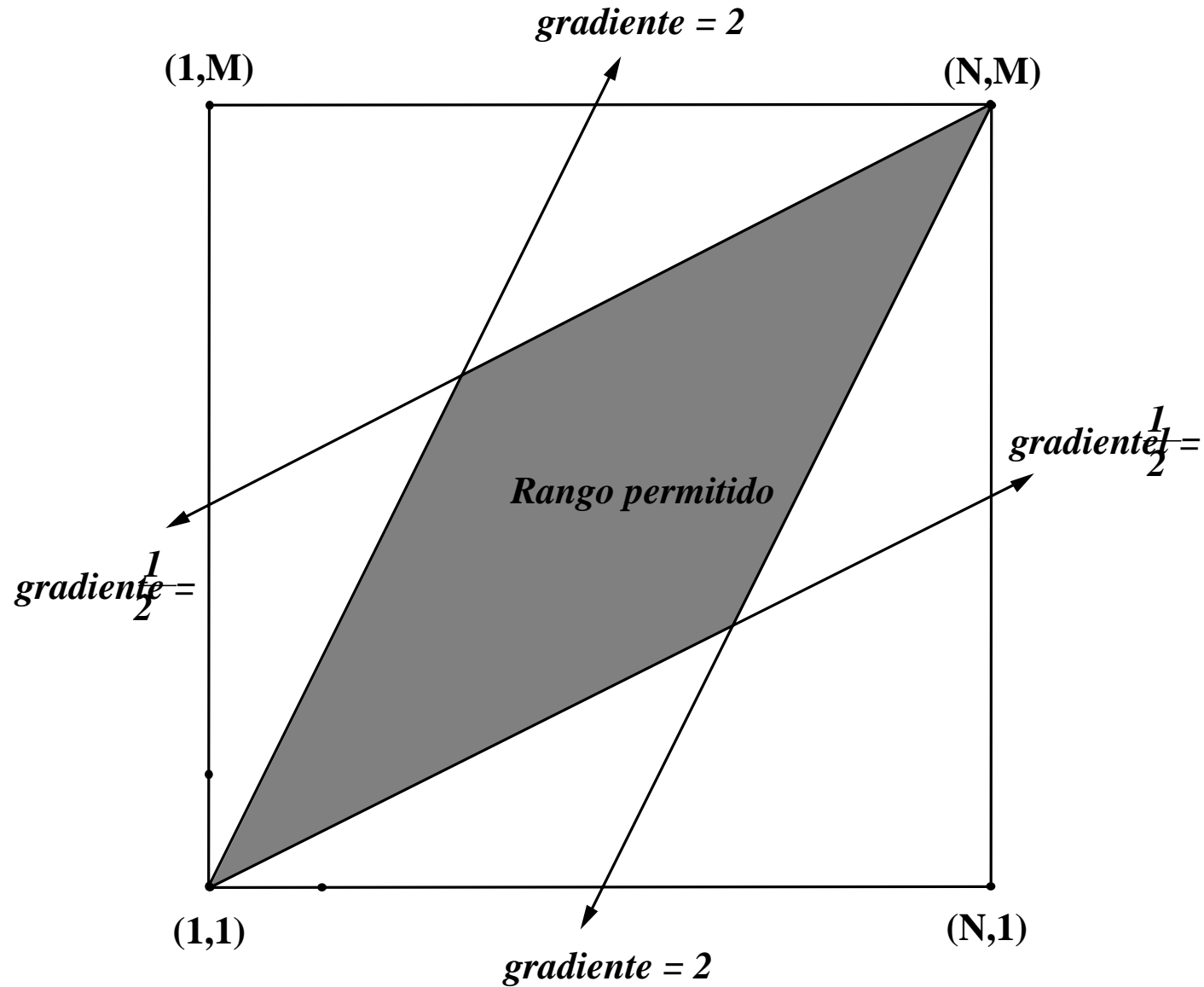
$$M_\phi = \sum_{k=1}^{K_\phi} m_k$$

# Cuestiones de DTW: Continuidad local



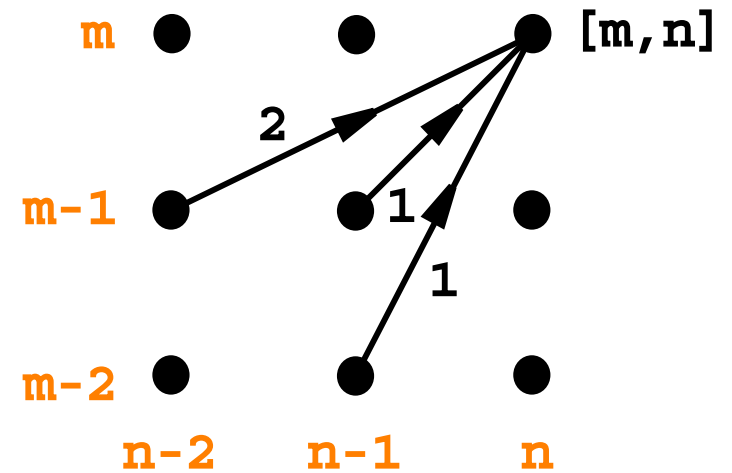
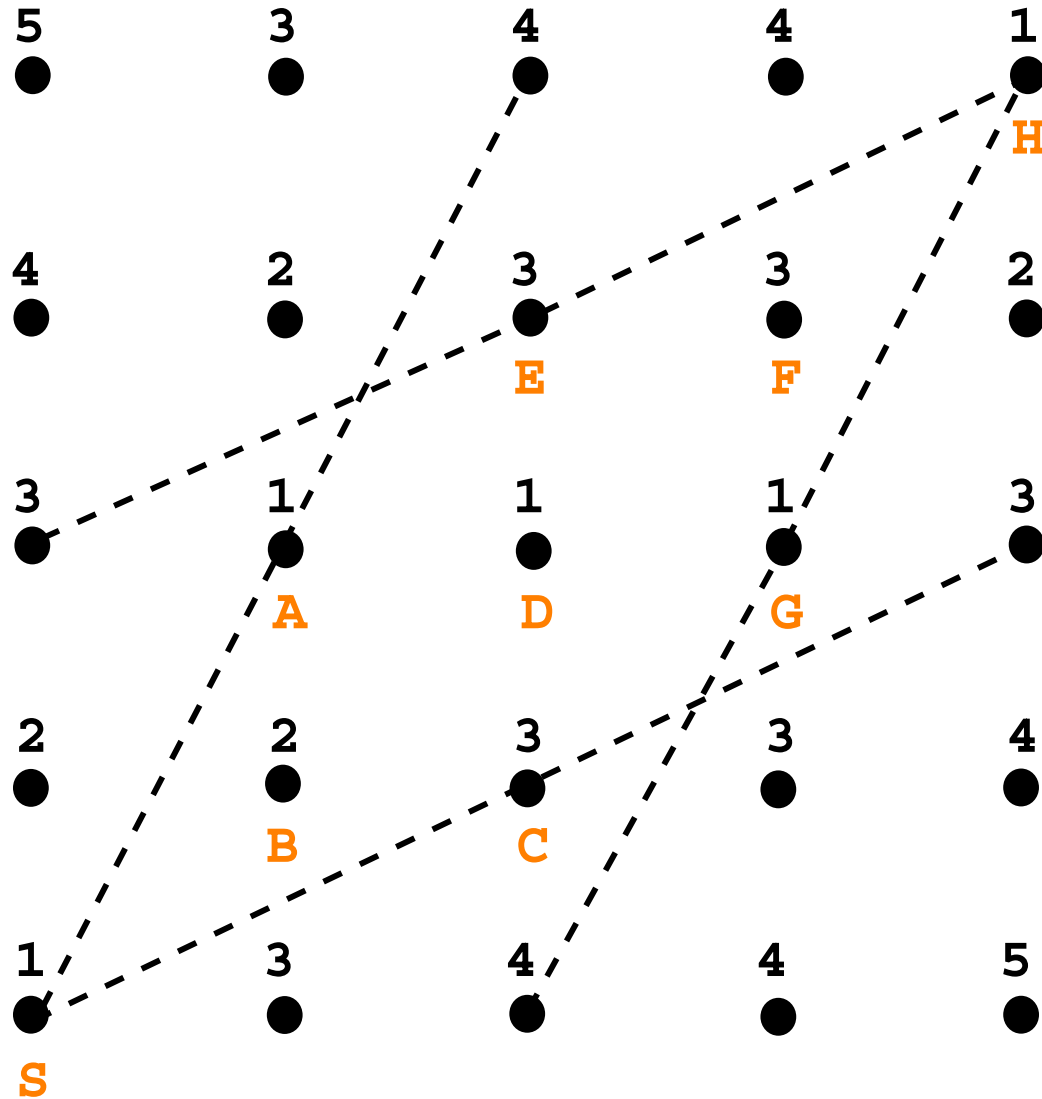
Las restricciones locales determinan la flexibilidad de alineamiento

# Cuestiones de DTW : Restricciones globales



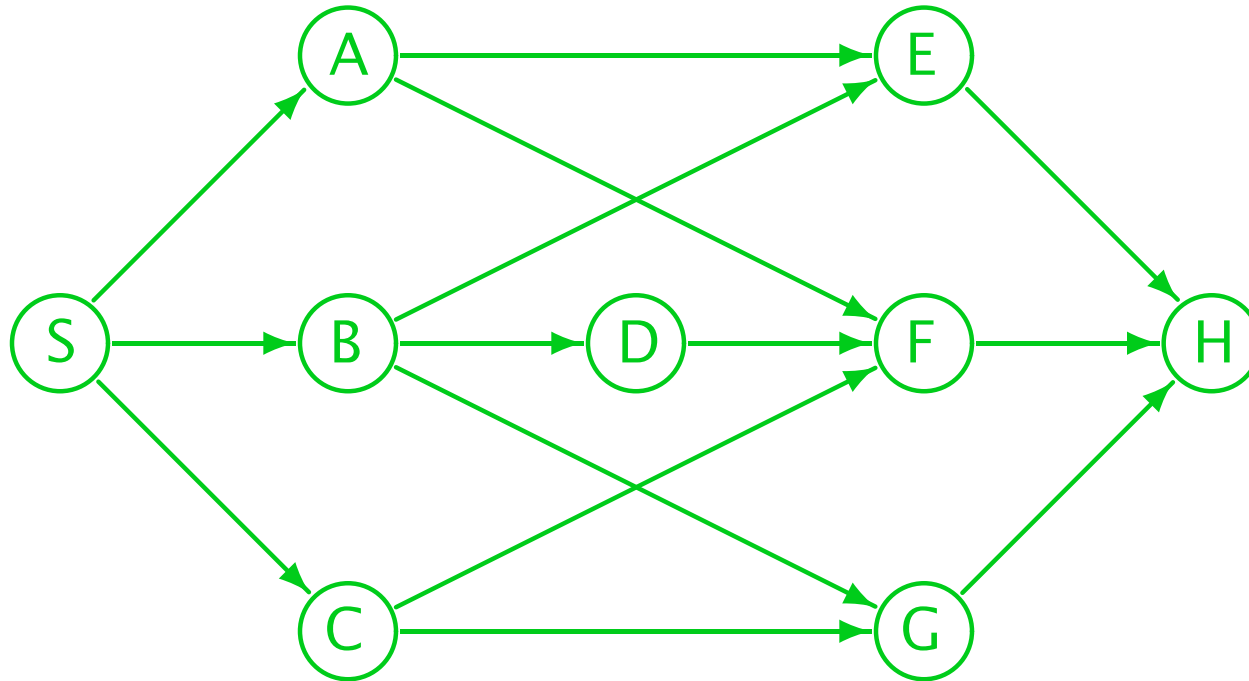
Las restricciones locales excluyen partes del espacio de búsqueda

# Cómputo de alineamiento de la DTW



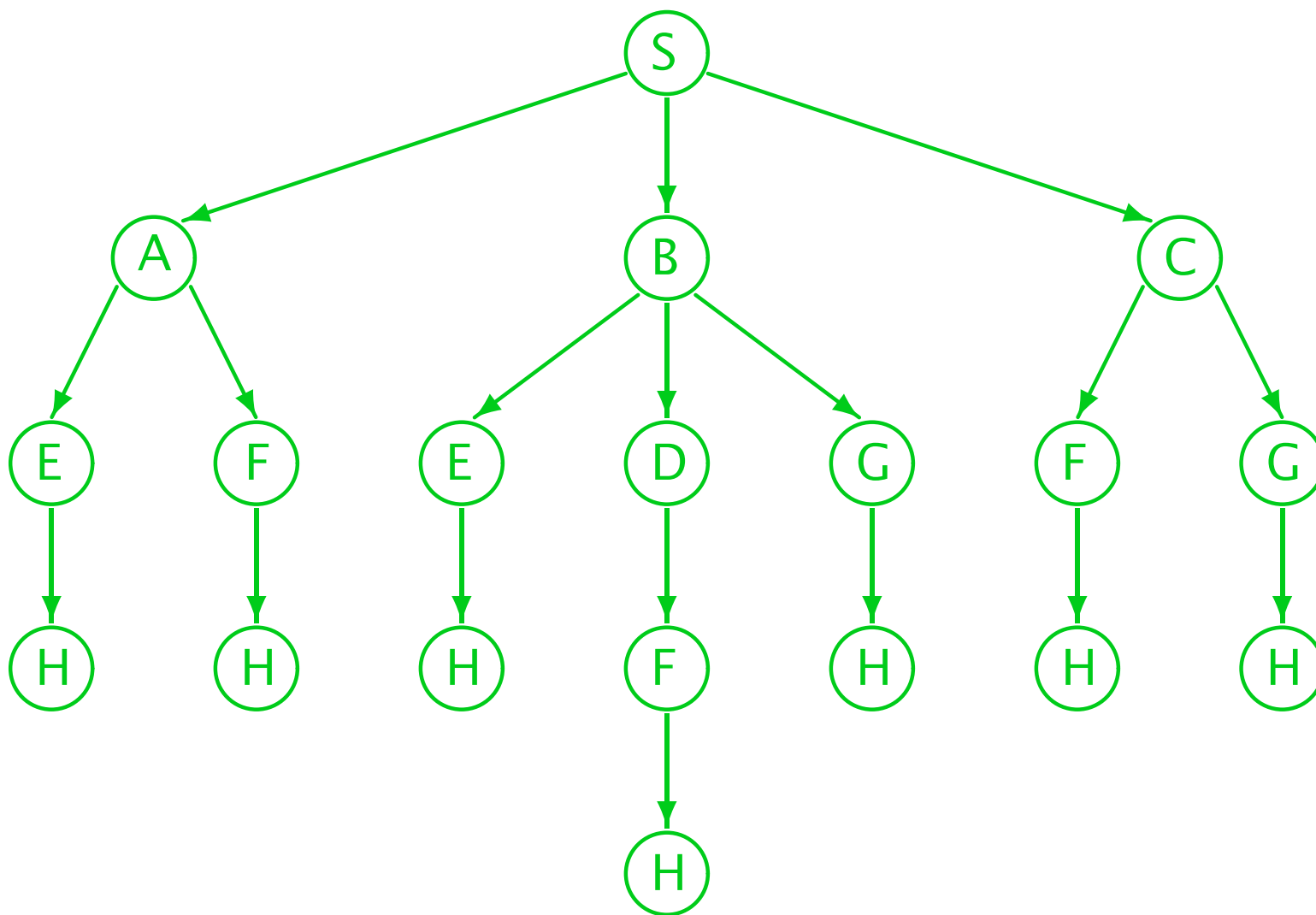
# Representaciones gráficas del espacio de búsqueda

- Los espacios de búsqueda pueden representarse como grafos dirigidos.



- Las trayectorias a través de un grafo pueden representarse mediante un árbol.

# Árbol del espacio de búsqueda



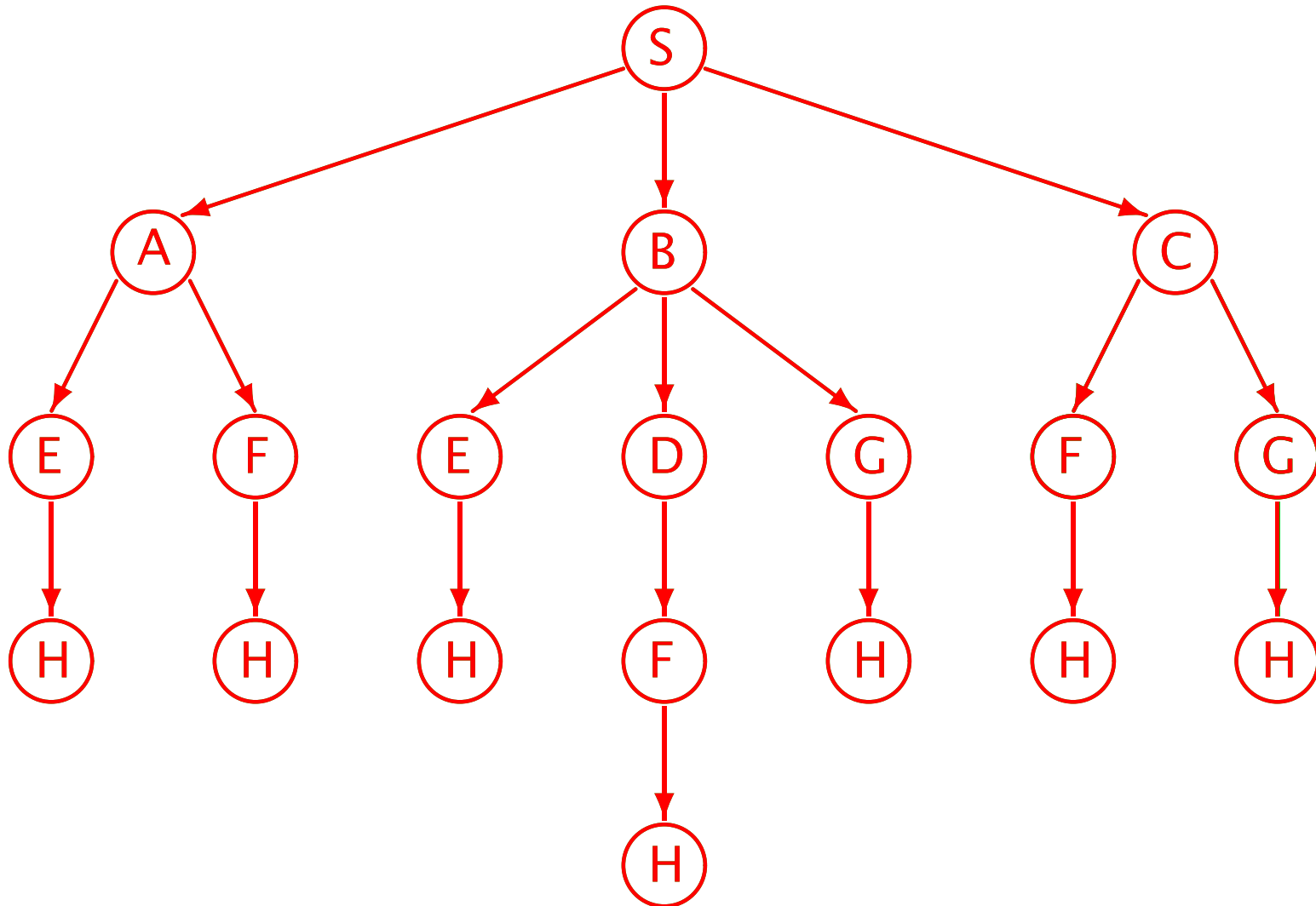
# Algoritmos de búsqueda gráfica

- Los métodos iterativos que utilizan una **cola** para almacenar las trayectorias parciales:
  - En cada iteración se extrae de la cola la trayectoria parcial superior y se amplía un nivel.
  - Las nuevas ampliaciones se vuelven a colocar en la cola.
  - La búsqueda se completa cuando se alcanza el objetivo.
- La profundidad de la cola está potencialmente ilimitada.
- Se pueden examinar los gráficos **pesados** con el fin de encontrar la **mejor** trayectoria.
- Los algoritmos **admisibles** garantizan la obtención de la mejor trayectoria.
- Muchos problemas de búsqueda relativos al habla, pueden configurarse para que procedan en tiempo síncrono.

# Búsqueda primero en profundidad

- Examina el espacio buscando una trayectoria al mismo tiempo.
- Las ampliaciones de la trayectoria se colocan **al principio** de la cola.
- La cola no se reordena ni se recorta.
- No muy apropiado para los espacios con trayectorias de extremos largos.
- Generalmente no se utiliza para encontrar la mejor trayectoria.

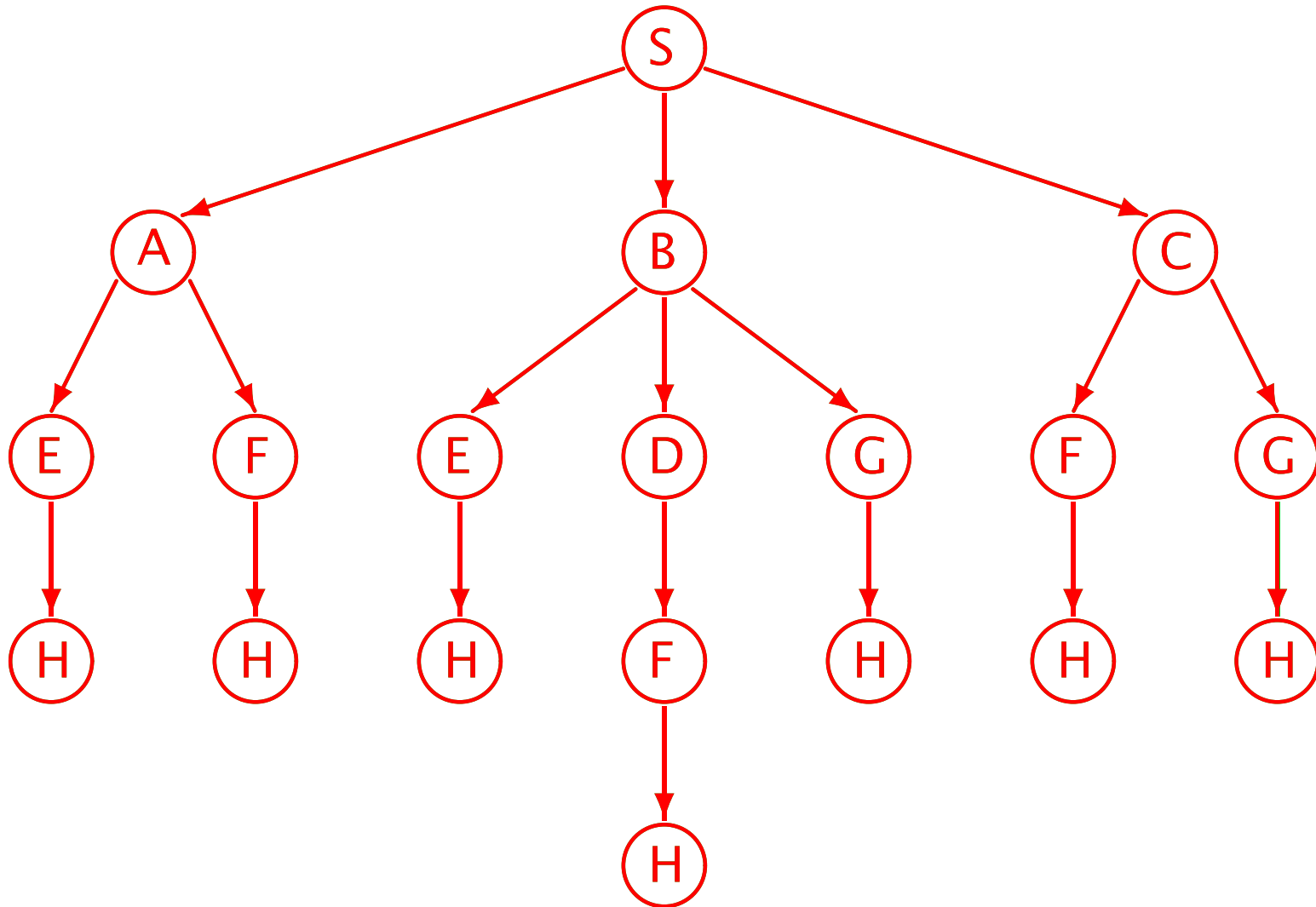
# Ejemplo de búsqueda primero en profundidad



# Búsqueda primero en amplitud

- Examina el espacio buscando todas las trayectorias en paralelo.
- Las ampliaciones de la trayectoria se colocan **al final** de la cola.
- La cola no se reordena ni se recorta.
- La cola puede ir aumentando rápidamente en espacios con muchas trayectorias.
- No se utiliza generalmente para encontrar la mejor trayectoria.
- Es posible potenciar su efectividad mediante el recorte.

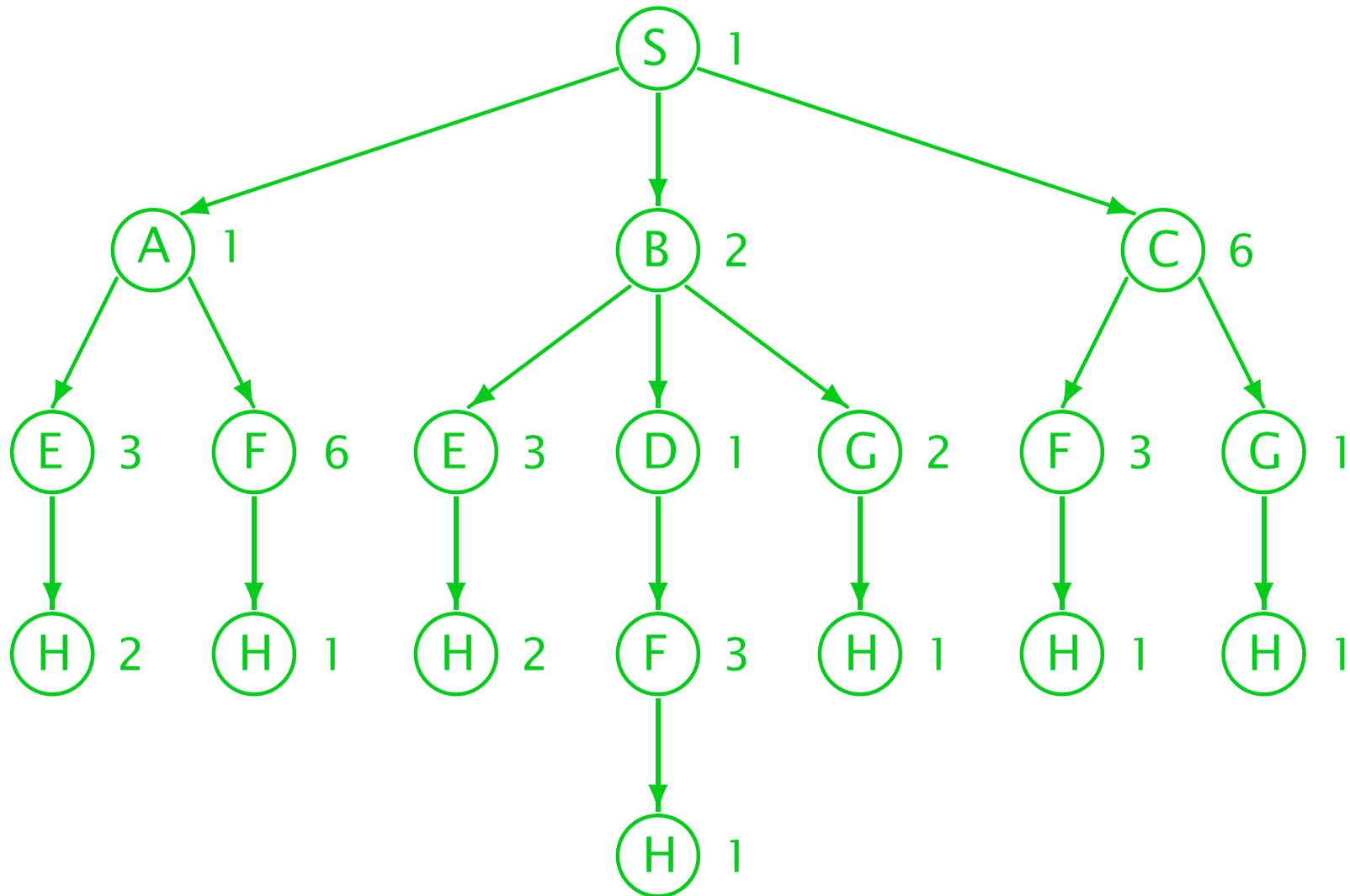
# Ejemplo de búsqueda primero en amplitud



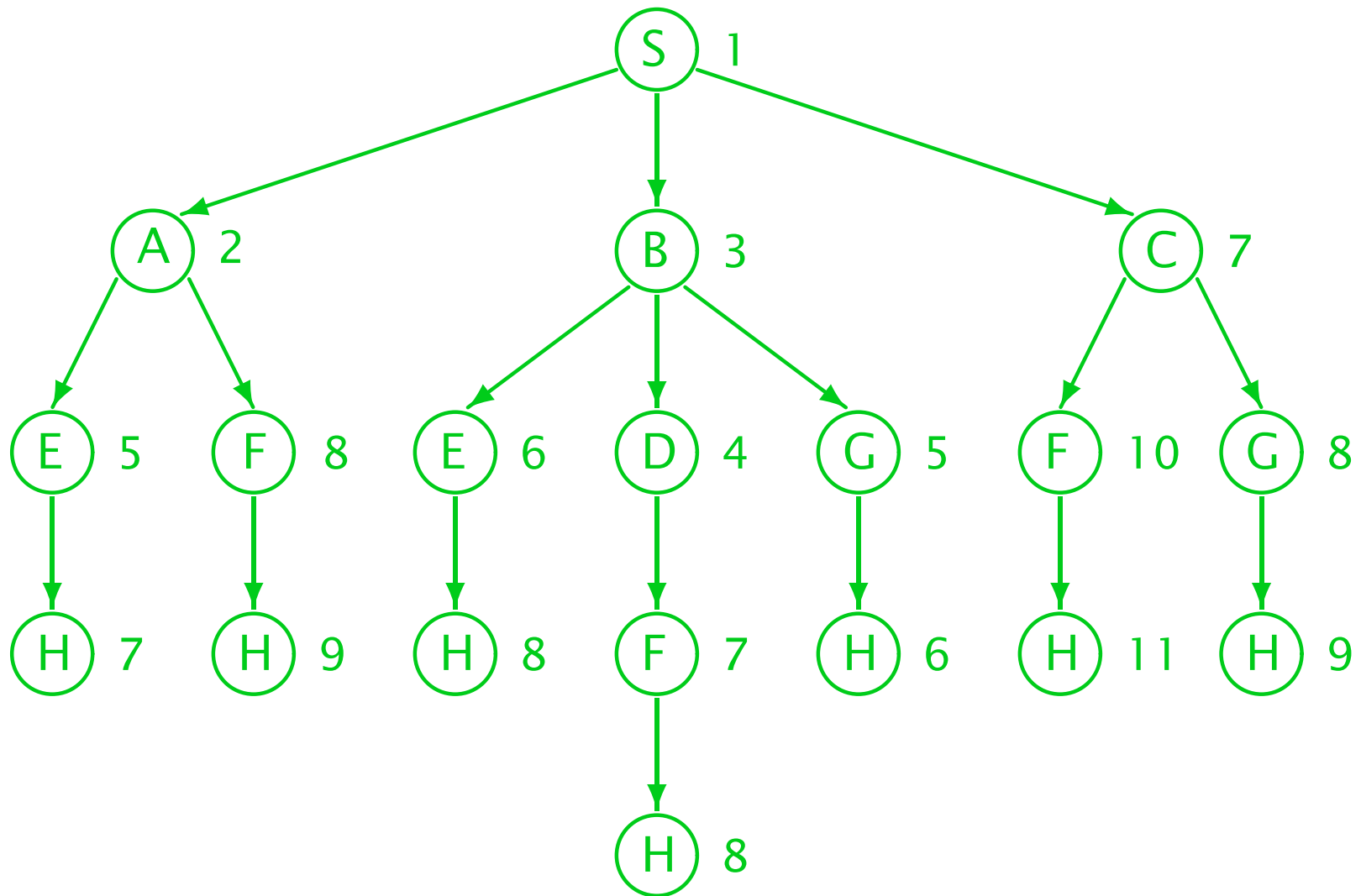
# Búsqueda primero el mejor

- Se emplea para examinar un gráfico **pesado**.
- Utiliza el **avance rápido** o el criterio **óptimo paso a paso**, según el cual cada iteración amplía la mejor trayectoria en curso.
- En cada iteración, la cola es **recurrida** según la puntuación acumulativa de cada trayectoria parcial.
- Si las puntuaciones de la trayectoria exhiben una conducta monotonía (ej.  $d(\mathbf{t}_i, \mathbf{r}_j) \geq 0$ ), la búsqueda puede acabar cuando una trayectoria completa presente una puntuación mejor que todas las trayectorias parciales activas.

# Representación arbórea (con puntuaciones de nodo)



## Representación arbórea (con puntuacionea acumulativas)





# Recorte de trayectorias parciales

- Tanto los algoritmos de avance rápido como los de programación dinámica, pueden obtener ventaja de la **subestructura óptima**:
  - Sea  $\phi(i, j)$  la mejor trayectoria entre los nodos  $i$  y  $j$
  - Si  $k$  es un nodo en  $\phi(i, j)$ :

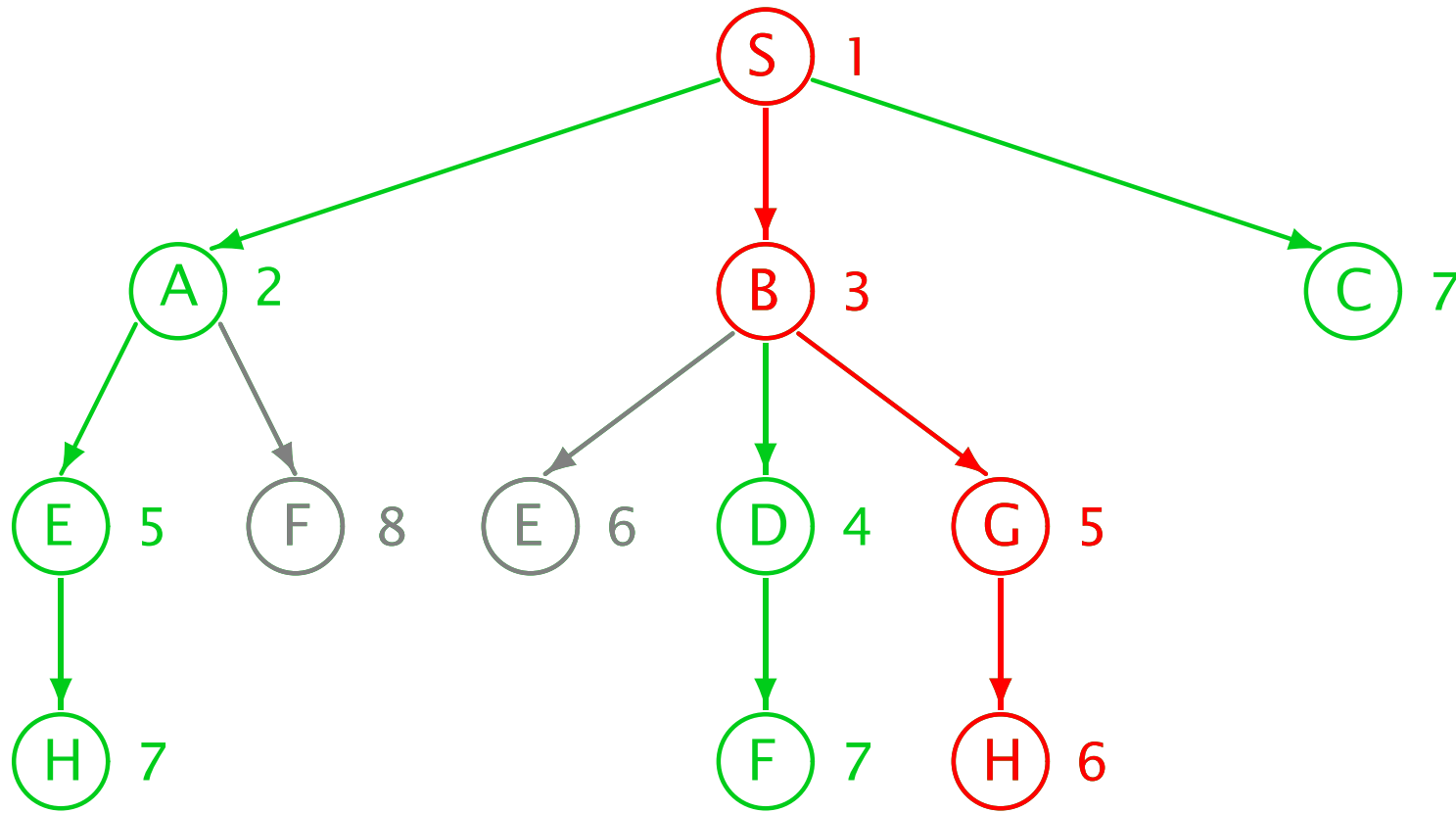
$$\phi(i, j) = \{\phi(i, k), \phi(k, j)\}$$

- Sea  $\varphi(i, j)$  el **coste** de  $\phi(i, j)$

$$\varphi(i, j) = \min_k (\varphi(i, k) + \varphi(k, j))$$

- El cálculo de las soluciones a los subproblemas sólo es necesario una vez.
- Las trayectorias parciales subóptimas se pueden descartar mientras se mantenga la admisibilidad de la búsqueda.

# Primera búsqueda mejor con recorte



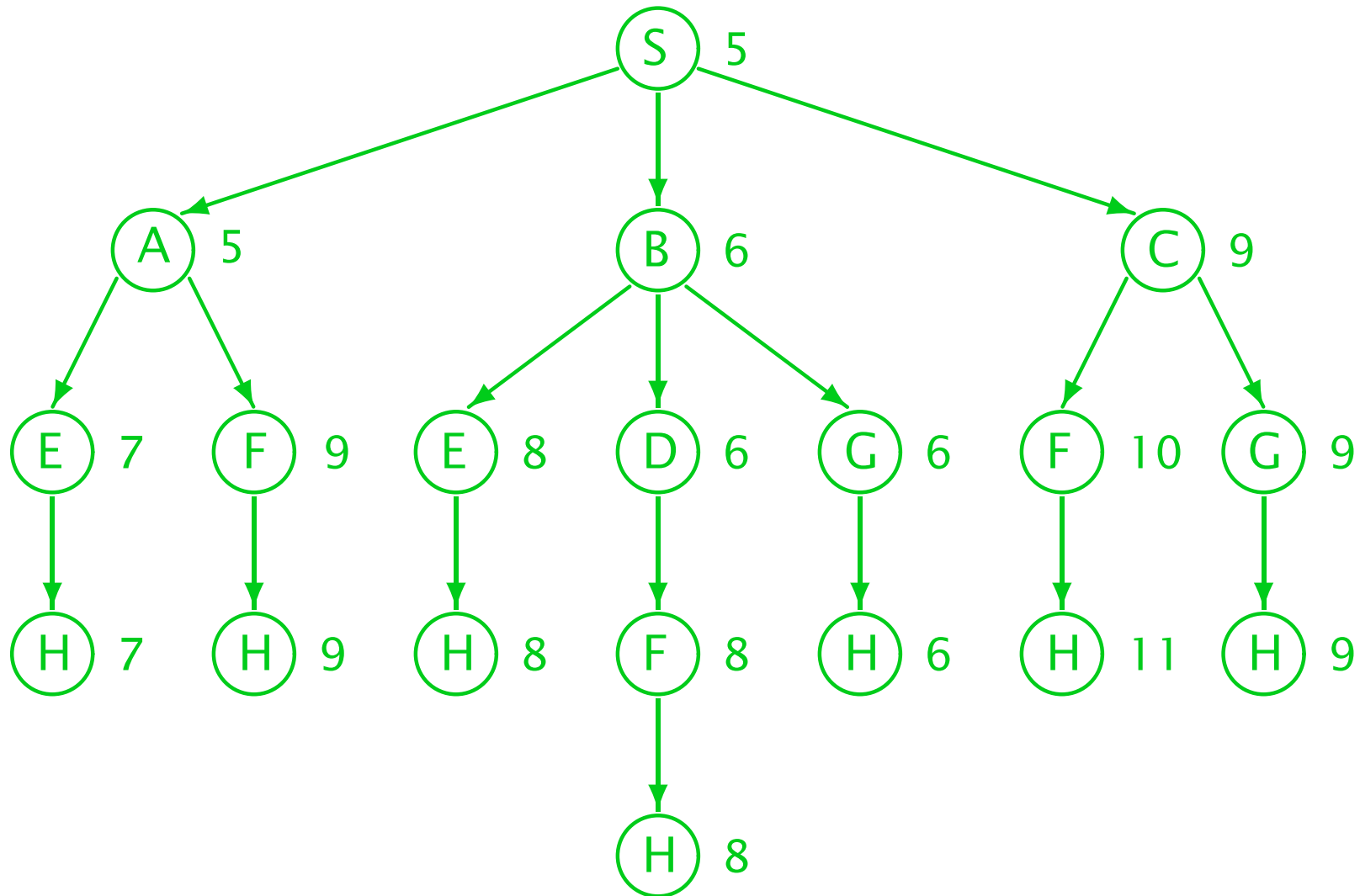
# Cálculo de puntuaciones futuras

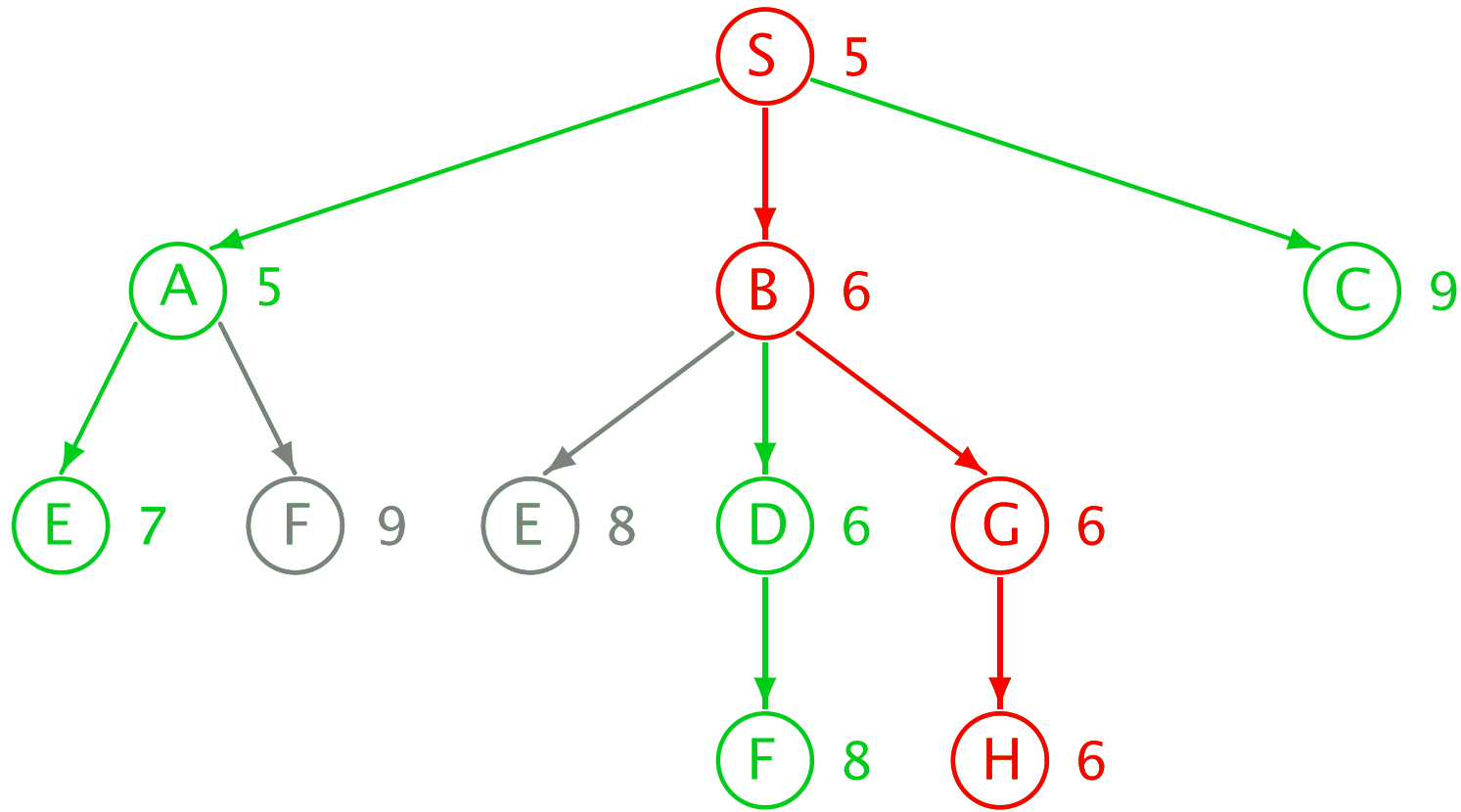
- Las puntuaciones de trayectoria parcial  $\varphi(1, i)$ , se pueden incrementar con cálculos futuros,  $\varphi(i)$ , del coste sobrante.

$$\varphi_{\phi} = \varphi(1, i) + \hat{\varphi}(i)$$

- Si  $\hat{\varphi}(i)$  es un cálculo demasiado bajo del coste sobrante, las trayectorias adicionales pueden recortarse mientras se mantenga la admisibilidad de búsqueda.
- Usos de la búsqueda  $A^*$ :
  - Estrategia de búsqueda el primero mejor
  - Recorte
  - Estimaciones futuras

# Representaciones arbóreas (con estimaciones futuras)

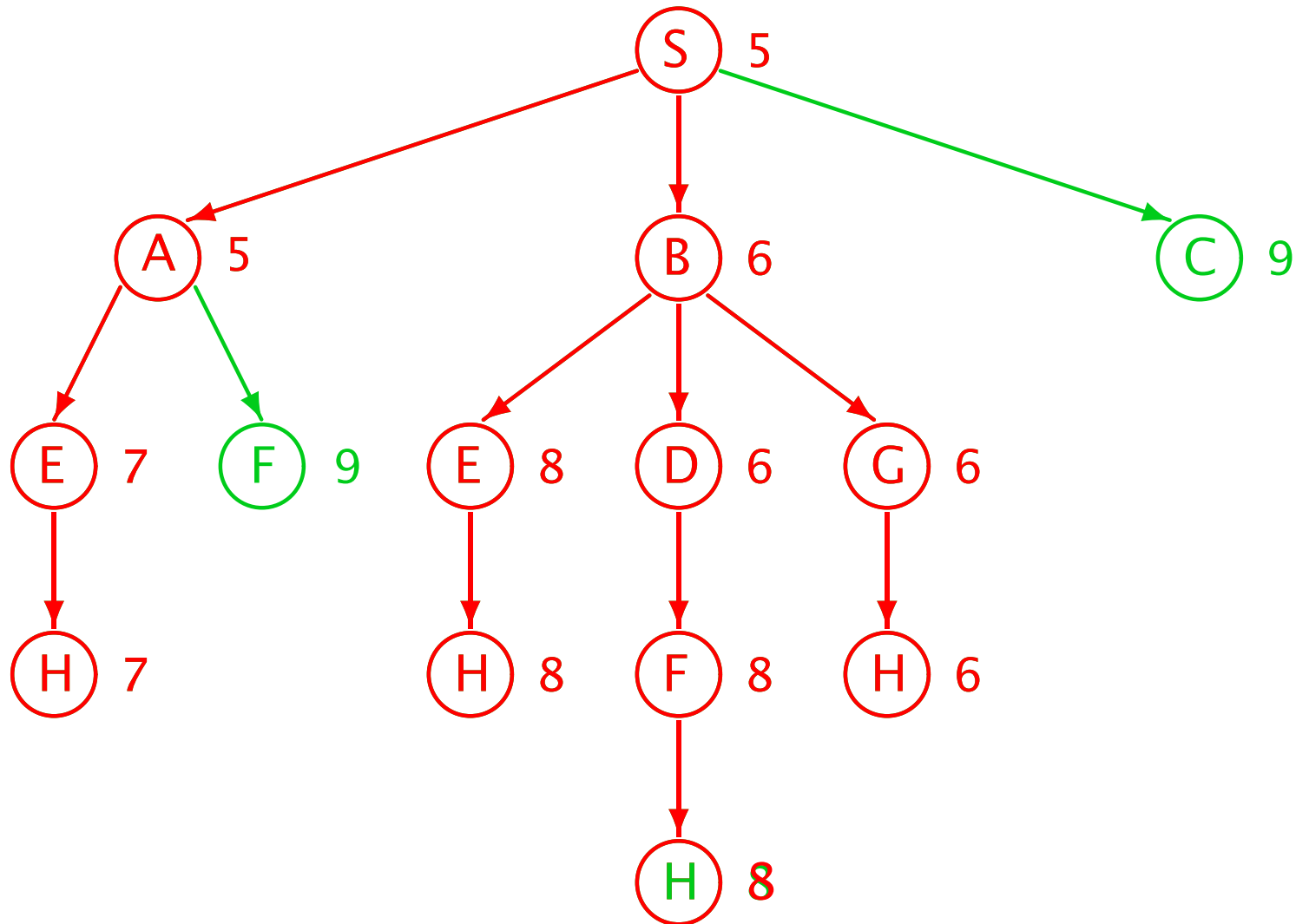


Ejemplo de búsqueda  $A^*$ 

## Búsqueda *N-Best* (N-mejor)

- Utilizada para computar trayectorias superiores  $N$ 
  - Puede ser repuntuada por técnicas más sofisticadas.
  - Empleada típicamente a nivel oracional.
- Puede utilizar búsqueda  $A^*$  modificada para clasificar trayectorias.
  - No se produce recorte de las trayectorias parciales.
  - Las trayectorias completadas se quitan de la cola.
  - Puede utilizar un umbral para recortar trayectorias, y todavía identificar las violaciones de admisibilidad.
  - También puede emplearse para generar un gráfico.
- Se pueden utilizar métodos alternativos para calcular las salidas N-mejores (ej., DP (programación dinámica) asíncrona).

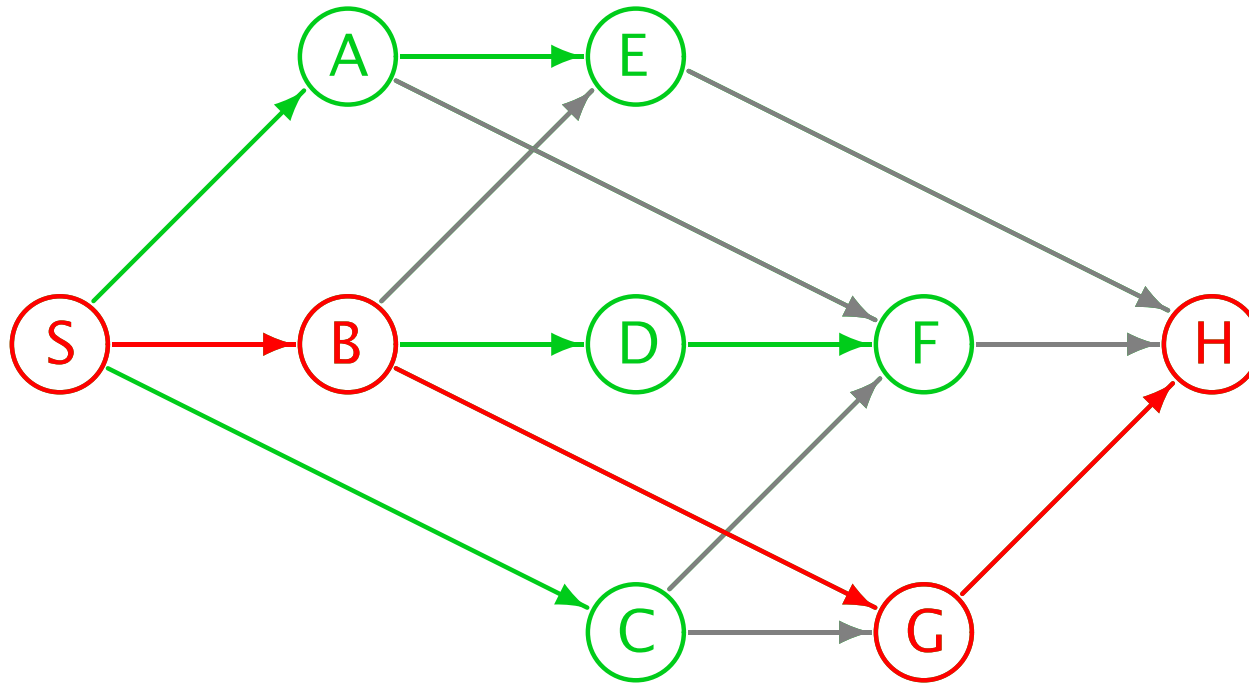
# Ejemplo de búsqueda N-mejor



# Programación dinámica (DP)

- Los algoritmos de DP no emplean la estrategia de avance rápido.
- Los algoritmos de DP normalmente aprovechan la subestructura óptima y los subproblemas de solapamiento, organizando la búsqueda de manera que el subproblema se resuelva sólo una vez.
- Se pueden implementar eficientemente:
  - El nodo  $j$  retiene sólo el coste de la mejor trayectoria de todos los  $\varphi(i, j)$
  - El mejor nodo id anterior es necesario para recuperar la mejor trayectoria.
- Puede ser en tiempo síncrono o asíncrono.
- DTW y Viterbi son búsquedas en tiempo síncrono y parecen de primero en amplitud con recorte.

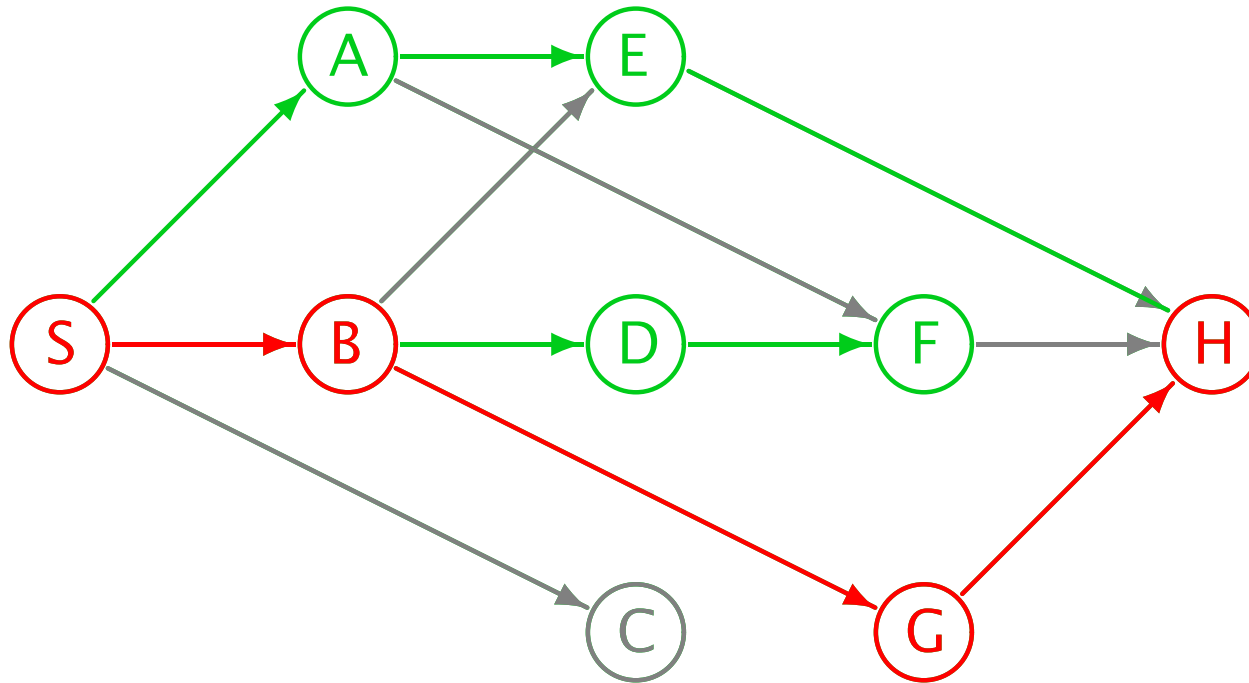
# Ejemplo de DP en tiempo síncrono



## Variaciones de búsqueda

- Puede utilizar un **ancho de haz** para reducir las hipótesis actuales.
  - Un ancho de haz puede ser estático o dinámico basado en la puntuación relativa.
- Puede utilizar una aproximación a un límite inferior en  $A^*$  de cara a la computación de la  $N$ -mejor.
- La búsqueda es inadmisibile, pero en la práctica puede resultar útil.

# Ejemplo de búsqueda en haz



- Cormen, Leiserson, Rivest, *Introduction to Algorithms*, 2ª ed., MIT Press, 2001.
- Huang, Acero y Hon, *Spoken Language Processing*, Prentice-Hall, 2001.
- Jelinek, *Statistical Methods for Speech Recognition*. MIT Press, 1997.
- Winston, *Artificial Intelligence*, 3ª ed., Addison-Wesley, 1993.