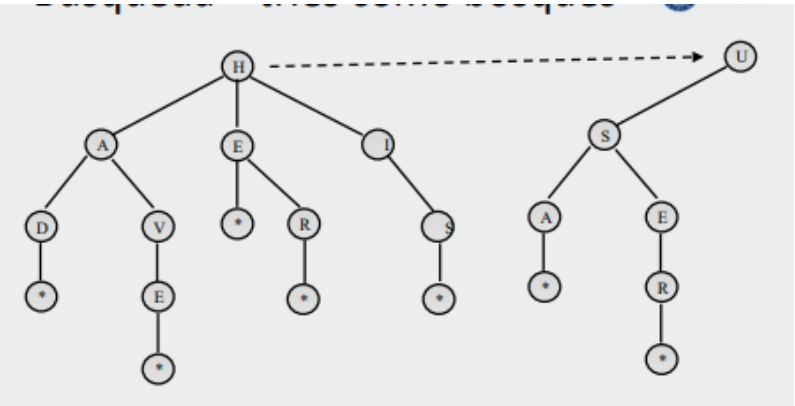
Tries

• Grado: número de subárboles de un nodo.

El orden de los hijos de un nodo interno de T está determinado por el orden canónico del alfabeto σ.

Ejemplo:



Acá “HAD” es la primera palabra, antes de “HE” la cual se encuentra después alfabéticamente hablando.

Usos de Tries:

- Enfoque adecuado cuando se realiza una serie de consultas sobre un texto fijo.
- Estructura apropiada para almacenar cadenas para soportar comparación de patrones rápida (proporcional al tamaño del patrón).
- En una aplicación de recuperación de la información,(ej.: secuencia de ADN en una base de datos genómica), la entrada es una colección S de strings, definidas mediante un alfabeto determinado

Operaciones primarias:
– Comparación de patrones (strings o palabras completas)
– Comparación de prefijos (dada una string X de entrada, encontrar todas las strings S que contienen a X como prefijo).

• Importante: ninguna string en S es prefijo de otra string.

– Esto asegura que cada string de S está asociada en forma única con un nodo externo de T.
–
Siempre se puede satisfacer esta condición, agregando un carácter especial al final de cada string. (ej: “*”)

Dos operaciones principales:

1. Búsqueda de palabras completas
    a. se desea determinar si un patrón dado está en una de las palabras del texto, exactamente.
    b. Ejemplo: índice, al encontrar la palabra queremos indicar las páginas del libro en que se encuentra
2. Búsqueda de prefijos
    a. Determina si un patrón dado es prefijo de alguna palabra almacenada en el trie
    b. Es la base para implementar funcionalidades como autocompletado y sugerencias
    c. Permite encontrar eficientemente todas las palabras que comparten un prefijo común

Búsqueda en tries

En la clase NodoTrie, el método buscar(string unaPalabra) devuelve el nodo que corresponde al último carácter del argumento, o el nodo nulo si ese argumento no está en el TRIE.

```
Comienzo
nodoActual ← this
Para cada carácter car de unaPalabra hacer
   unHijo ← nodoActual.obtenerHijo(car) //depende de la estructura del nodo
   Si unHijo = nulo entonces
      devolver nulo
   Sino
      nodoActual ← unHijo
   fin si
fin para cada

Si nodoActual.esFinDePalabra entonces
   devolver nodoActual
sino
   devolver nulo
fin si
Fin
```

Trie comprimido – “patricia”

• ventajoso sólo cuando se lo utiliza como una estructura de índice auxiliar

• colección de strings que ya está almacenada

• no se le requiere que almacene realmente todos los caracteres

A la hora de hacer una búsqueda si no es exitoso (es decir, si no encuentro la palabra), el orden no es m, sino m*d, siendo m el tamaño de la palabra a buscar y d el tamaño del alfabeto. Esto al menos dentro de un trie estándar.

El nivel de un nodo es el mismo que la cantidad de letras que forman la palabra que va desde la raíz hasta ese nodo, ej: Si yo tengo la palabra “Caracol” y marco que hay una palabra hasta “cara” , entonces el nodo “a” (refiriéndome a la segunda a) sería nivel 4.