PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN
IIC2233 - PROGRAMACIÓN AVANZADA

Ayudantía 06

2^o semestre 2018
Septiembre 2018

Estructuras de Datos: Árboles y listas ligadas

Basada en la AC 04 2017-2

Introducción

En esta ayudantía deberán implementar un **ContacTrie**, el cual es tipo especial de Trie. El Trie es un árbol utilizado para almacenar palabras de manera eficiente. Un ejemplo de un Trie es el que se visualiza continuación[1], el cual posee las palabras "dia", "dime", "mama" y "mia".

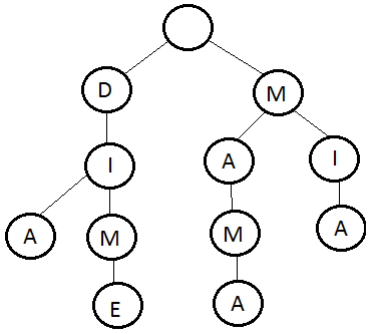

Figura 1: Trie con palabras "dia", "dime", "mama" y "mia"

Un Trie comienza con sólo la raíz, la cual no posee una letra propia. Cuando se ingresa una nueva palabra, se ve si algún nodo hijo del actual (en este caso la raíz) posee la primera letra de la palabra. Si ese nodo no existe, se crea. Luego, se pasa a ese nodo y se realiza lo mismo con el resto de la palabra.

Por ejemplo, a continuación veremos cómo se ingresa la palabra "dias" a un Trie vacío.

Figura 2: El Trie comienza con sólo la raíz

[1]Fuente: https://www.tecnohobby.net

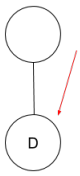

Figura 3: Se agrega un nodo hijo con la primera letra de “dias”

Como pueden ver se creó el nodo con la letra “D”. Ahora se debe continuar ingresando el resto de la palabra (quedan las letras “ias”).

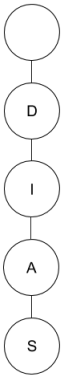

Figura 4: Se puede apreciar el Trie con la palabra “dias”

Si ahora se quisiera ingresar la palabra “mias”, se tendría que crear toda una nueva rama, ya que si bien comparten la mayoría de las letras, las primeras letras de las palabras son distintas.

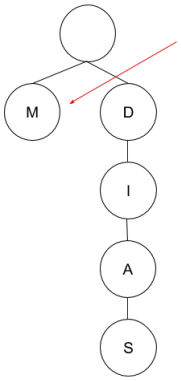

Figura 5: Como no hay un nodo con la letra “M” en la raíz, se crea uno.

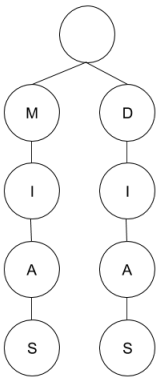

Figura 6: Trie con las palabras “mias” y “dias”

Por otro lado, si se quisiera agregar la palabra “diana”, sí se puede aprovechar de nodos ya existentes. Esto porque la raíz ya posee un nodo hijo con la letra “D”.

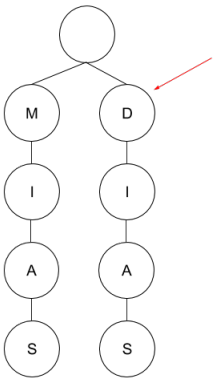

Figura 7: La raíz de este Trie ya posee un nodo con la letra “D”

De la misma forma puede avanzar hasta llegar al nodo con la letra “A”, el cual no posee un nodo hijo con la letra “N”. Por lo tanto, debe crear una nueva rama para poder ingresar el resto de la palabra.

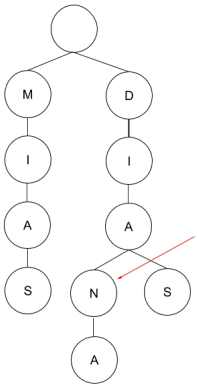

Figura 8: Trie con las palabras “mias”, “diana” y “dias”

Como pueden notar en cualquier nodo hijo (u hoja) se almacena una letra. Sin embargo, se puede almacenar más información en cada nodo. Para esta ayudantía deberán implementar un `ContacTrie` que almacene contactos, es decir, palabras asociadas a números. A continuación se explicarán los métodos que debe tener su ContacTrie.

- `add_contact`: Este método permite agregar contactos entregándole un string con el nombre y un entero con el número. Antes de agregar un nuevo contacto se debe verificar que éste sea válido. Primero deben revisar que el string sólo posea letras (puede tener tildes), ya sean mayúsculas o minúsculas. Si el string entregado posee algún otro caracter o es vacío se debe entregar el mensaje "Nombre Inválido". Si el string es válido, debe ser transformado a mayúsculas. Luego, deben verificar que el número entero entregado sea mayor a 0. Si el número entregado no es entero, o es menor o igual a 0, se debe imprimir el mensaje "Número Inválido". Si los argumentos son válidos pero el nombre del contacto ya existe en el ContacTrie, éste no se debe sobreescribir, en cambio, se debe entregar el mensaje "Contacto ya Existe". Si la operación tiene éxito se debe imprimir el mensaje "Contacto agregado con éxito".

- `change_contact_number`: Este método debe permitir cambiar el número de un contacto existente. Para esto se le debe entregar un string con el nombre del contacto y un número entero para reemplazar al número existente. Para esto se debe comprobar que tanto el string o el número sean válidos. Si no lo son se debe abortar el cambio y entregar el mensaje "Argumentos Inválidos". Si el string es válido, debe ser transformado a mayúsculas. Si el string y el entero son válidos pero el contacto no existe, se debe imprimir "Contacto Inexistente" (y detener la operación). Si el cambio es válido, se debe cambiar el número por el nuevo e imprimir "Cambio realizado con éxito".

- `ask_for_contact`: Este método debe consultar por un número de un contacto. Para esto se le debe entregar un string con el nombre del contacto pedido. Si el string es inválido se debe imprimir el mensaje "Nombre Inválido". Si el string es válido, debe ser transformado a mayúsculas. Si el contacto no existe se debe imprimir "Contacto Inexistente". Si el contacto existe se debe imprimir el nombre y el número de la siguiente forma "`({nombre}, {número})`".

- `get_all_contacts`: Este método no recibe argumentos y debe retornar una lista con todos los contactos. Cada contacto en la lista debe estar contenido en una tupla de la forma (nombre, número). El orden en que se muestran los contactos es irrelevante.

- `merge_tries`: Este método recibe como argumento a otro objeto de la clase ContacTrie. Si este objeto no es un ContacTrie, se debe imprimir el mensaje "Debe ser un ContacTrie". Si lo es, se deben agregar todos sus contactos al ContacTrie original, saltándose aquellos que ya existen en él, es decir, no se sobreescriben los contactos ya existentes. Una vez terminada la operación se debe imprimir en pantalla "Unión Exitosa".

- **sumar** dos instancias de ContacTries con el operador `+` generando una nueva instancia. Esta nueva instancia debe poseer todos los contactos del primero, unido con los contactos del segundo. Es decir, si se realiza la operación `trie1 + trie2`, el trie generado debe poseer todos los contactos del trie1, agregándole, luego, los contactos del trie2 que no posee. En el caso de que el segundo objeto no sea de clase ContacTrie, se debe imprimir el mensaje "Operación No Soportada" y retornar None.

Importante: Sólo los nodos con la letra final de cada palabra almacenen el número. Por ejemplo, supongamos que el siguiente Trie tiene los contactos “Hernan” y “Flor” con los números 9876 y 1234 respectivamente.

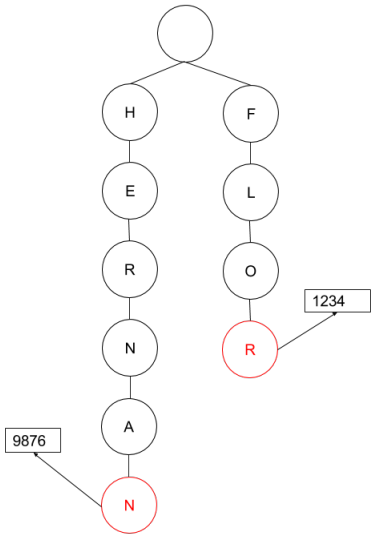

Figura 9: Las hojas de color rojo almacenan un número

De esta manera los nodos intermedios quedan “libres” para almacenar un número. Así, se puede agregar un contacto con nombre “flo” con el número 43567 sin tener que alterar la estructura del ContacTrie.

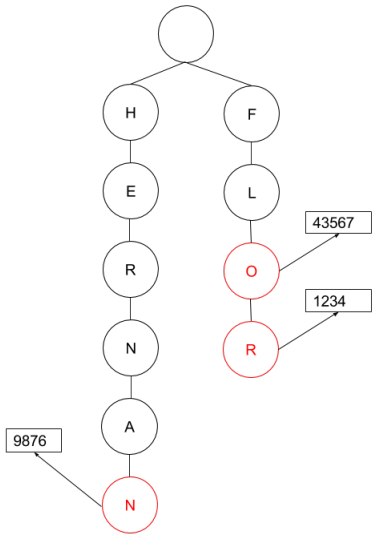

Figura 10: Se le agrega un número al nodo con la letra “O”

En resumen, el ContacTrie que deben implementar debe poseer los siguientes métodos y características:

- `add_contact(name, number)`: Recibe una palabra (`str`) y un número (`int`), y añade al ContacTrie esa palabra asignándole el número telefónico.
- `change_contact_number(name, number)`: Recibe un contacto (nombre) y un número, y actualiza el número telefónico de dicho contacto.
- `ask_for_contact(name)`: Recibe una palabra y debe imprimirla junto con su número de teléfono, en una tupla de la forma "`(name, number)`".
- `get_all_contacts()`: Retorna una lista con todos los contactos del ContacTrie, en tuplas de la forma `[(name_1, number_1), ... , (name_n, number_n)]`.
- `merge_tries(other_trie)`: Recibe otro trie, y debe agregar todos los contactos del other_trie que no posea.
- Si se suman 2 tries (Ej: `trie3 = trie1 + trie2`), el resultado debe ser una nueva instancia de un ContacTrie, que contenga todos los contactos del primer trie unido con los del segundo.