Autómata Traductor de Código Morse

Ornella Ivonne Benzi Juncos
ornellabenzi@hotmail.com

Cátedra Principios de Computadoras II
Profesor: Mg. Ricardo Coppo
rcoppo@uns.edu.ar
Dpto. Ingeniería Eléctrica y de Computadoras
Universidad Nacional del Sur
Av. Alem 1253 - Primer piso
8000 Bahía Blanca
Tel:/ FAX 291 459 5154 / 291 459 5100 ext. 3301

Resumen El trabajo consiste en el desarrollo de un software que implementa dos autómatas, uno capaz de traducir caracteres en código morse al alfabeto latino y otro que realiza la operación inversa, implementando una estructura de datos enlazada eficiente. Además, el programa emula el sonido del telégrafo en función del código Morse obtenido a través de la reproducción de archivos de audio previamente generados con la duración y frecuencia estándar del código.

1. Introducción

El código Morse [1], también conocido como alfabeto morse o clave morse, es un código o sistema de comunicación que permite la comunicación telegráfica a través de la transmisión de impulsos eléctricos de longitudes diversas o por medios visuales, como luz, sonoros o mecánicos. Este código consta de una serie de puntos, rayas y espacios, que al ser combinados entre sí pueden formar palabras, números y otros símbolos.
Actualmente, debido a su confiabilidad, el código Morse se sigue utilizando en la navegación marítima y aérea.

1.1. Objetivos del trabajo

Se pretende implementar un traductor bidireccional para convertir cadenas en ambos alfabetos implementando un tipo de datos abstracto basado en dos autómatas finitos traductores y emular el sonido del telégrafo. La estrategia utilizada puede utilizarse para traducción de códigos más complejos en escenarios donde la eficiencia sea mandatoria.

2. Modelo

2.1. Autómata Finito

Un autómata finito o máquina de estados, es un modelo matemático de la computación. Es una máquina abstracta que puede estar en exactamente

un estado, de un número finito de ellos en cualquier tiempo dado. El autómata finito puede cambiar de un estado a otro en respuesta a alguna entrada externa; el cambio de un estado a otro se llama transición. Un autómata finito está definido por el listado de sus estados, su estado inicial y las condiciones para cada transición.

2.2. Autómata Traductor

Un autómata traductor [2] es aquel capaz de generar una salida basada en una entrada dada y/o un estado usando acciones. Son usados para aplicaciones de control y en el campo de la lingüística computacional.
Un autómata finito traductor es una séxtupla $(\Sigma, \Gamma, S, s_0, \delta, \omega)$ donde:

- Σ es el alfabeto de entrada (un conjunto de símbolos finito no vacío)
- Γ es el alfabeto de salida
- S es el conjunto de estados
- s_0 es el estado inicial, un elemento de S
- δ es la función de transición: $\delta : S \times \Sigma \rightarrow S$
- ω es la función de salida

Si la función de salida es una función de un estado y del alfabeto de entrada ($\omega : S \times \Sigma \rightarrow \Gamma$) esa definición corresponde al modelo Mealy y puede modelarse como una Máquina de Mealy. Si la entrada depende solo del estado ($\omega \rightarrow \Gamma$) esa definición corresponde al modelo Moore y puede modelarse como una Máquina de Moore.

Modelo utilizado En este trabajo se utilizaron dos autómatas traductores, los cuales se caracterizan continuación.

Traductor Morse a Alfabeto latino

- El alfabeto de entrada Σ={'.', '-', '#' , '##' } ; donde # representa el espacio.
- El alfabeto de salida Γ = { 'A'..'Z' , '#'} ; donde # representa el espacio.
- S es el conjunto de estados, uno por cada letra y un estado inicial.
- s_0 es el estado inicial, un elemento de S.
- Función de transición: $\delta : S \times \Sigma \rightarrow S$, las transiciones son posibles ingresando punto, raya o el espacio.
- ω es la función de salida que devuelve la letra correspondiente cuando se ingresa un espacio.

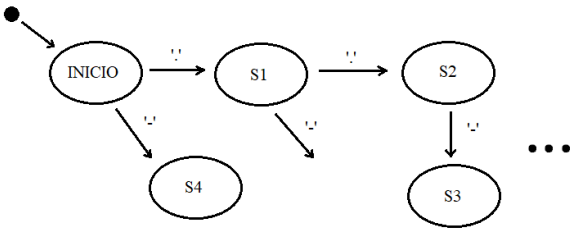

Figura 1. Diagrama de estados aproximado del autómata traductor Morse-Letra

Traductor Alfabeto Latino a Morse

- Σ= 'A'..'Z', '#' ; donde # representa el espacio.
- Γ = '.', '-', '#', '##' ; donde # representa el espacio.
- S es el conjunto de estados, uno correspondiente a cada letra y un estado inicial.
- s_0 es el estado inicial, un elemento de S
- δ es la función de transición: $\delta : S \times \Sigma \rightarrow S$, las transiciones son
- ω es la función de salida, la cual devuelve el código morse correspondiente

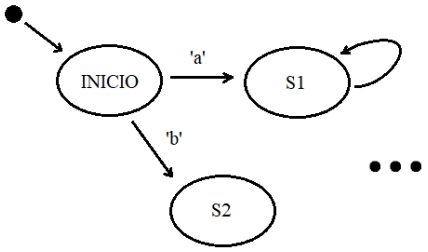

Figura 2. Diagrama de estados aproximado del autómata traductor Letra-Morse

3. Implementación

Para la programación de las estructuras de datos utilizadas para la implementación de ambos autómatas traductores se utilizó C++. Para la interfaz gráfica se utilizó Qt, que es un framework multiplataforma orientado a objetos de software libre y código abierto ampliamente usado para desarrollar programas que utilicen interfaz gráfica de usuario.

3.1. Estructura de datos

Árboles Binarios Un árbol [3] es una estructura que modela una estructura jerárquica, con un valor en la raíz y subárboles con un nodo padre, representado como un conjunto de nodos enlazados. Cada subárbol posee una arista única que lo une con la raíz.

Un árbol binario [3] es una estructura de datos de árbol en la cual cada nodo tiene como máximo dos hijos, cada uno etiquetado como hijo izquierdo o hijo derecho. Si un nodo no tiene hijos entonces este es llamado un nodo hoja o nodo externo. Ejemplo en la Figura 3.

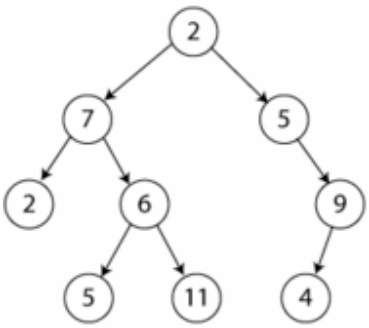

Figura 3. Ejemplo de árbol binario

Estructura de datos implementada Para la implementación del traductor de Morse a Alfabeto latino se utilizó un árbol binario en cuyos nodos se almacenan las letras del abecedario y el camino hacia cada nodo representa el código morse del carácter.

Para la traducción inversa se utilizó una estructura auxiliar que constó en un arreglo de punteros a nodo, en el cual se almacenaron punteros a cada uno de los nodos del árbol para evitar la búsqueda en el árbol. Las Figuras 4 y 5 ilustran las estructuras de datos de cada traductor respectivamente.

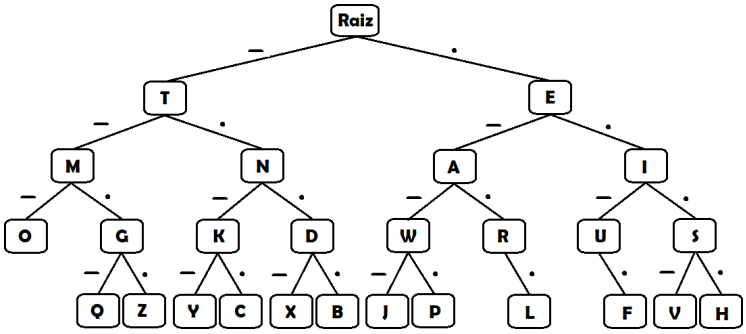

Figura 4. Estructura de datos autómata traductor Morse-Letra

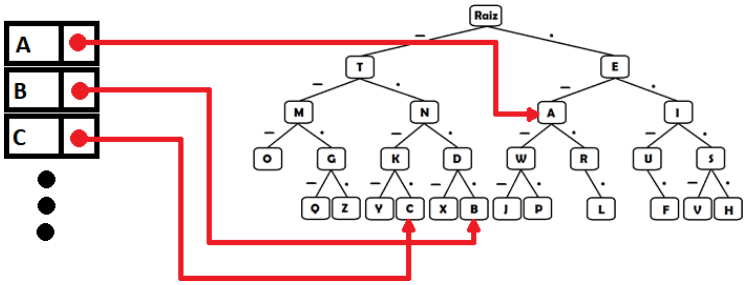

Figura 5. Estructura de datos autómata traductor Letra-Morse

Ejemplo de traducción Para ilustrar el uso de los autómatas anteriormente mencionados, supongamos que se quiere obtener la traducción de la cadena "• • • – – – • • •", el autómata comienza en el estado inicial, la raíz del árbol y, dado que el primer caracter es un punto, pasa al estado que corresponde al subárbol izquierdo. Luego, desde ese nuevo nodo, dependiendo de si el siguiente caracter es un punto o una raya, pasa al subárbol izquierdo o derecho, respectivamente. Cuando el caracter es un espacio, el autómata pasa al estado final y devuelve la traducción del caracter, que es el que está almacenado en el nodo en el que está al momento de recibir el espacio.

3.2. Diseño de clases y UML

En la Figura 6 se detalla el UML del programa. En él podemos ver las relaciones entre las clases utilizadas y su cardinalidad.

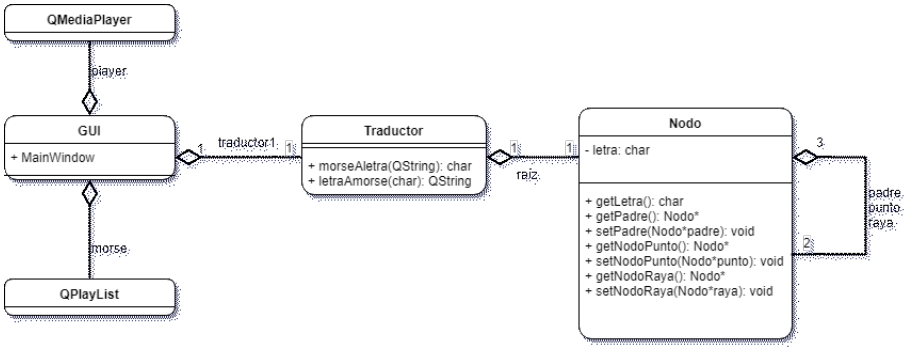

Figura 6. UML del programa

3.3. Interfaz gráfica

El entorno gráfico utilizado se realizó a través del UI de Qt [4]. Se dispusieron dos LineEdit, uno para la cadena alfabética y otro para la cadena en morse, con sus respectivos botones para traducir de uno a otro. Además, se agregó un botón para la reproducción del sonido, un botón de "Ayuda" que despliega un MessageBox con las letras del abecedario y su equivalencia en código morse, y una imagen ilustrativa de un telégrafo.

Figura 7. Interfaz gráfica del programa

Validación de entrada Para evitar errores en la ejecución del programa, se resolvió restringir valores de entrada a los LineEdit reciben el código morse o la cadena de alfabeto a ser traducidos, desde la interfaz gráfica. Para ello, se utilizaron validadores (QValidator [6]) basados en expresiones regulares de los caracteres deseados.

Expresiones regulares Las expresiones regulares [5] son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto.

Ejemplos:
"[.| |-]*" representa "punto o espacio o guion medio"
"[a-zA-Z|]*" representa "letras entre 'a' y 'z' y entre 'A' y 'Z' o espacio"

Sonido En la implementación del programa se incluyó la simulación sonora de la cadena en código morse. Se generó una Playlist dinámica en función de la cadena de entrada utilizando las clases QMediaPlayer y QPlayList [7] provistas por Qt [4].
Los sonidos fueron generados por el sitio *Wavtones* [8] respetando la relación estándar entre las duraciones de puntos, rayas y silencios.

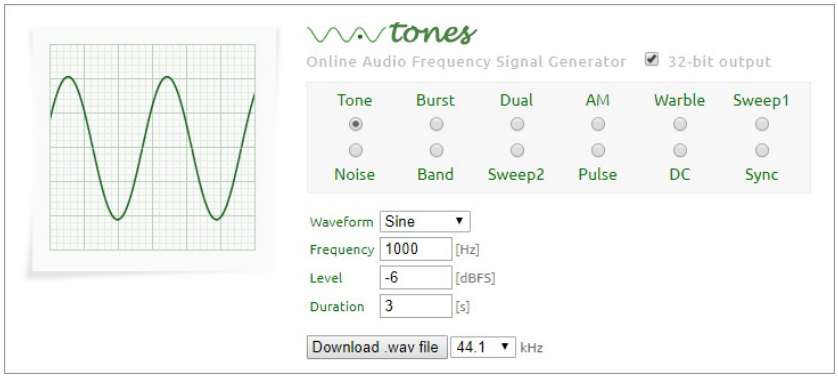

Figura 8. Captura del sitio web utilizado para generar los archivos de audio

4. Conclusión

La implementación de un árbol binario para realizar el traductor resultó sumamente eficiente a la hora de traducir código morse a alfabeto latino, no así para el proceso inverso, dado que requirió de punteros que guardaran la posición de cada nodo para así desencriptar de abajo hacia arriba el código correspondiente al carácter latino ingresado.

Referencias

1. “Morse Code”, Wikipedia, https://en.wikipedia.org/wiki/Morse_code
2. “Finite State Machine”, Wikipedia, https://en.wikipedia.org/wiki/Finite-state_machine
3. Coppo, Ricardo. "Principios de Computadoras II - Apuntes de Cátedra”
4. “Qt (Software)”, Wikipedia, https://en.wikipedia.org/wiki/Qt_(software)
5. “Expresión Regular”, Wikipedia, https://es.wikipedia.org/wiki/Expresion_regular
6. Qt Documentation “QValidator”, doc.qt.io/
7. Qt Documentation “QMediaPlayer” y “QPlayList” , doc.qt.io/
8. Wavtones, http://www.wavtones.com/functiongenerator.php