Memoria Práctica `Lex`
Modelos de Computación

Joaquín Avilés de la Fuente Arturo Olivares Martos

27 de diciembre de 2024

Índice

Resumen

En la presente memoria se detallará la práctica llevada a cabo en la asignatura de Modelos de Computación del tercer curso del Doble Grado en Ingeniería Informática y Matemáticas de la Universidad de Granada. En ella, hemos implementado un menú con diversas funciones para analizar un texto introducido por el usuario, el cual detallaremos más adelante en las distintas secciones de la memoria.

1. Introducción

En nuestro caso, para no limitarnos a un único caso de uso, lo cual probablemente implicaría usar menos expresiones regulares, hemos decidido implementar un menú con diversas opciones que permiten al usuario analizar un texto introducido por él mismo. Esto proporciona así más versatilidad, a la vez que nos facilita la implementación de un mayor número de expresiones regulares.

La funcionalidad del programa se puede probar de forma directa mediante el menú creado, o se puede tambiar probar por separado con cada programa. Además, hemos creado un archivo `makefile` que permite que la ejecución sea más sencilla.

Todos estos aspectos los explicaremos de forma detallada en cada una de las secciones de este artículo.

Observación. Todos las rutas relativas que se proporcionan suponen que nos encontramos en la carpeta raíz de esta práctica.

2. Comprobación de Números de Teléfono

Este analizador se encuentra en el archivo `regex_tfno.l`. Para compilarlo, se debe ejecutar el siguiente comando:

```
$ flex++ -o regex_tfno.cpp regex_tfno.l
$ g++ -Wall -o regex_tfno regex_tfno.cpp -lfl
```

Para ejecutarlo, se debe ejecutar el siguiente comando:

```
$ ./regex_tfno <fichero>
```

donde `<fichero>` es el archivo que se desea analizar en busca de números de teléfono. De forma alternativa, se recomienda usar el `makefile` proporcionado usando la siguiente regla:

```
$ make telefonos
```

En este caso, el archivo de texto que se analizará es `Telefonos/telefonos.txt`, por lo que deberá almacenarse en ese archivo las cadenas que se deseen procesar.

Este programa analiza el archivo pasado en búsqueda de números de teléfono. Por un lado, informa de las cadenas que no han sido identificadas con números de teléfono de ningún país registrado y, por otro lado, de los números de teléfono válidos que se han encontrado, informa en qué país son válidos. Actualmente tan solo se ha implementado la detección de números de teléfono españoles, puesto que son los que conocemos de primera mano, pero la extensión a otros países es directa. Para detectar los números de teléfono, se han usado las siguientes expresiones regulares:

- `tfno_espanol_grupos3`:

  ```
  ((00|"+")34(" ")?)?([0-9]{3}(" ")?){3}
  ```

```
arturoolvrs@arturoolvrs-MacBookAir:~/Desktop/Lex-Practica$ make telefonos
flex++   Telefonos/regex_tfno.l
mv lex.yy.cc Telefonos/regex_tfno.cpp
g++   -std=c++11 -g -Wall   Telefonos/regex_tfno.cpp -o Telefonos/regex_tfno_exe -lfl
./Telefonos/regex_tfno_exe Telefonos/telefonos.txt


Cadenas leidas no identificadas:
        - 12345678912
        - abcdefgh

Números por Pais:
- España:
        - +34 888 88 88 88
        - 0034 999999999
        - 643 333 333
        - 999 999 997

arturoolvrs@arturoolvrs-MacBookAir:~/Desktop/Lex-Practica$
```

Figura 1: Ejemplo de ejecución del programa `regex_tfno`.

En este caso, se da la posibilidad a que el número de teléfono esté agrupado en grupos de 3 cifras, separados por un espacio. En primer lugar, puede comenzar con el prefijo internacional (`34` en el caso de España), que ha de ir precedido de `00` o `+`. A continuación, se encuentras los tres grupos de 3 cifras, en total 9 cifras. Se permite que haya un espacio entre los grupos de cifras y tras el prefijo internacional.

- `tfno_espanol_grupos2`:

  ```
  ((00|"+")34(" ")?)?([0-9]{3}(" ")?){1}([0-9]{2}(" ")?){3}
  ```

  Esta expresión es similar, solo que permite que los 6 últimos finales se agrupen en tres grupos de 2 cifras, en lugar de dos grupos de 3 cifras. De nuevo, se permite que haya un espacio entre los grupos de cifras y tras el prefijo internacional (si lo hay).

Hemos limitado a estos casos, puesto que son los más habituales en España. No obstante, se podrían añadir más expresiones regulares para detectar otros formatos de números de teléfono. Tenemos por tanto la siguiente regla, la cual fuerza a que se cumpla una de las dos expresiones regulares anteriores, y esto sea lo único que se encuentre en la línea:

```
^{tfno_espanol_grupos3}|{tfno_espanol_grupos2}$   {tfno_pais["España"].insert(yytext);}
```

Como vemos, los teléfonos detectados los almacenamos en un conjunto asociado a la clave `España`, de forma que sería muy fácilmente generalizable a un mayor número de países. Un ejemplo de funcionamiento de este programa se encuentra en la Figura 1.