83GIIN-Compresión y Recuperación de Información Multimedia

Actividad 2.- Compresión sin pérdida

Objetivo

El propósito de esta actividad es reforzar los fundamentos de la compresión sin pérdida, centrándose en el factor de compresión, la entropía y la importancia de modelar una fuente para obtener resultados óptimos en la compresión. Para ello, se proporciona un archivo comprimido que incluye implementaciones en lenguaje C de los algoritmos de compresión Huffman, codificación aritmética y codificación basada en diccionarios (LZSS), junto con el conjunto de datos proporcionado en la primera actividad.

Descripción

En esta tarea, nos familiarizaremos con las implementaciones en C de tres técnicas de compresión sin pérdida. Estas implementaciones fueron obtenidas de tres repositorios mantenidos por Michael Dipperstein en GitHub:

- Codificación Huffman
- Codificación Aritmética
- Codificación basada en Diccionarios (lzss)

En los repositorios de Michael Dipperstein, se puede encontrar una explicación detallada de cada algoritmo y las consideraciones relevantes de su implementación.

Para facilitar la descarga, he preparado un archivo comprimido llamado "Actividad_2.zip", que contiene el directorio "Actividad_2". Dentro de este directorio, tenemos 5 subdirectorios y 4 archivos de comandos (scripts) (Figura 1).

```
drwxr-xr-x 4 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users 4096 Mar  8 13:44 arcode-master
drwxr-xr-x 5 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users 4096 Mar 12 10:02 huffman-master
drwxr-xr-x 2 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users 4096 Mar  8 13:24 InputData
drwxr-xr-x 4 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users 4096 Mar  8 13:44 lzss-master
drwxr-xr-x 2 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users 4096 Mar  8 13:25 OutputData
-rwxr-xr-x 1 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users  698 Mar  8 11:14 test_arcode.sh
-rwxr-xr-x 1 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users  248 Mar  8 12:32 test_huffman_code.sh
-rwxr-xr-x 1 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users  796 Mar  8 11:10 test_huffman.sh
-rwxr-xr-x 1 UNIVERSIDADVIU\angela.diserio2 UNIVERSIDADVIU\domain users  427 Mar  8 12:12 test_lzss.sh
```

Figura 1. Estructura de directorios

Subdirectorios (azul):

- **arcode-master**: implementación en C de la codificación aritmética.
- **huffman-master**: implementación en C de la codificación huffman.
- **InputData**: archivos de prueba que se usarán en la actividad.

- **lzss-master**: implementación en C de la codificación basada en diccionarios lzss.
- **OutputData**: carpeta en donde se almacenan los resultados de las ejecuciones.

Scripts (verde):

- test_arcode.sh
- test_huffman.sh
- test_huffman_code.sh
- test_lzss.sh

Los *scripts* o archivos de comandos están configurados para aplicar cada ejecutable de compresión sobre todos los archivos que se encuentran en el directorio **InputData**. Durante la ejecución de estos *scripts* se muestra el archivo procesado, el tamaño en bytes del archivo original y el tamaño del archivo comprimido. Además, verifica que la versión descomprimida del archivo sea igual al original. De existir alguna diferencia, mostrará las líneas diferentes.

En el caso del script **test_huffman_code.sh**, se genera como salida un archivo con la codificación canónica de cada símbolo de la fuente de datos analizada. En este caso no genera el archivo comprimido. Por tanto, para cada archivo del subdirectorio **InputData** se genera un archivo con la asignación de códigos canónicos que será salvado en **OutputData**.

Proceso

- Descomprimir 83GIIN_Act2.zip y posteriormente para cada algoritmo de compresión dado generar el ejecutable correspondiente.
- Para compilar y ejecutar programas implementados en lenguaje C en un entorno Windows, puedes utilizar un compilador C compatible con Windows, como MinGW (Minimalist GNU for Windows) o Cygwin:
    - MinGW: Descarga e instala MinGW desde su sitio web oficial (https://osdn.net/projects/mingw/releases/).
    - Cygwin: Descarga e instala Cygwin desde su sitio web oficial (https://www.cygwin.com/). Durante la instalación, asegúrate de seleccionar los paquetes relacionados con el compilador GCC.
- Una vez que dispongas del compilador C, debes ingresar a cada una de los directorios (huffman-master, arcode-master y lzss-master) y ejecutar el comando ***make***. En nuestro caso tomará como argumento el archivo ***Makefile*** que contiene las reglas necesarias para compilar los archivos fuentes en C y generar el ejecutable. En huffman-master se genera el ejecutable **huffman**, en arcode-master genera **arcode** y en lzss-master genera **lzss**.
- Una vez generados los ejecutables, para conocer los argumentos que se pueden usar en la línea de comando del ejecutable usamos la opción -? Junto al nombre del ejecutable:

  ./huffman -?

```
[UNIVERSIDADVIU\angela.diserio2@a-2g9g7sffhu9mt huffman-master]$ ./huffman -?
Usage: huffman <options>

options:
  -C : Encode/Decode using a canonical code.
  -c : Encode input file to output file.
  -d : Decode input file to output file.
  -t : Generate code tree for input file to output file.
  -i<filename> : Name of input file.
  -o<filename> : Name of output file.
  -h|?  : Print out command line options.
```

COMANDO	ACCIÓN
./huffman -c -i ecoli.fasta -o SALIDA	Comprime ecoli.fasta y el archivo comprimido se salva en SALIDA
./huffman -d -i SALIDA -o RECUPERADO	Descomprime SALIDA y guarda el resultado RECUPERADO
./huffman -C -c -i ecoli.fasta -o SALIDA	Comprime ecoli.fasta usando código canónico. Salva resultado en SALIDA
./huffman -C -d -i SALIDA -o RECUPERADO	Descomprime SALIDA usando código canónico. Salva resultado en RECUPERADO
./huffman -c -t -i ecoli.fasta -o CODIGOS	Salva en CODIGOS la tabla de códigos canónicos de los símbolos de la fuente de datos

Archivos de datos

En el subdirectorio InputData tenemos disponibles los archivos tanto de texto plano como las imágenes en formato RAW de la primera actividad.

Para cada archivo:

1. Comprimir y descomprimir cada archivo usando los algoritmos de compresión dados. Podéis usar los archivos de comandos (.sh) incluidos en la actividad.
2. Calcular la tasa de compresión que se obtiene para cada una de las técnicas de compresión.
3. Mostrar los resultados mediante una tabla como se indica a continuación:

Nombre Archivo	Tamaño original (bytes)	Entropía calculada (bits)	Longitud promedio a partir de los códigos Canónicos (bits)	Huffman		Aritmético		Lzss	
				Bytes	Tasa de Compresión	Bytes	Tasa de Compresión	Bytes	Tasa de Compresión

En la tabla se debe mostrar el nombre del archivo analizado, su tamaño original, la entropía calculada a partir del archivo original, la **longitud promedio en bits de cada símbolo cuando se usa huffman canónico**, y para cada una de las técnicas de compresión se indica el tamaño del archivo comprimido en bytes y la tasa de compresión que se obtiene.

Finalmente, realizar una comparación crítica de los resultados obtenidos usando las codificaciones Huffman, aritmética y lzss tomando en cuenta las características del archivo analizado.

Detalles sobre la entrega

La entrega consistirá en un informe técnico en donde se muestran los resultados de las actividades realizadas sobre cada uno de los archivos análisis de los resultados y conclusiones.

La entrega se hará a través del Campus VIU en un archivo pdf

Fecha de entrega: 31 de marzo de 2025

Rúbrica para la evaluación

	Niveles de rendimiento			
Criterios	**Excelente 9-10**	**Notable 7-8**	**Aprobado 5-6**	**Suspenso 0-4**
Dominio de los conceptos **Ponderación 40,00%**	**100,00 %** Identifica y explica adecuadamente todos los conceptos relacionados con el tema	**80,00 %** Identifica y explica adecuadamente la mayoría de los conceptos relacionados con el tema	**60,00 %** Identifica y explica adecuadamente alguno de los conceptos relacionados con el tema	**40,00 %** Identifica y explica muy poco o ninguno de los conceptos relacionados con el tema
Relación entre Teoría de Información y compresión **Ponderación 30,00%**	**100,00 %** Identifica y relaciona correctamente los conceptos de la teoría de la información con la compresión de datos, razón de compresión, etc.	**80,00 %** Identifica y relaciona correctamente los conceptos de la teoría de la información con la compresión de datos, razón de compresión, etc.	**60,00 %** Identifica y relaciona algunos de los conceptos de la teoría de la información con la compresión de datos, razón de compresión, etc.	**40,00 %** Identifica y relaciona pocos o ninguno de los conceptos de la teoría de la información con la compresión de datos, razón de compresión, etc.
Estructura, presentación y redacción del informe **Ponderación 10,00%**	**100,00 %** El informe presenta una muy buena estructura con todos los apartados. Aplica correctamente las reglas gramaticales, ortográficas y de sintáxis	**80,00 %** El informe presenta una buena estructura pero falta algún apartado importante. Presenta algunos errores de redacción y gramaticales	**60,00 %** El informe carece de varios apartados importantes. Presenta muchos errores de redacción y ortográficos.	**40,00 %** El trabajo no presenta una estructura de reporte técnico. Se cometen múltiples errores gramaticales
Resultados **Ponderación 20,00%**	**100,00 %** Los resultados obtenidos del análisis son correctos	**80,00 %** Unos pocos resultados del análisis no son correctos	**60,00 %** Una gran parte de los resultados no son correctos	**40,00 %** Los resultados son incorrectos