Compresión/Convolución

Integrantes:

Jhiens Angel Guerrero Ccompi - 20210147J

Jose Rodolfo Estacio Sanchez - 20214027A

Joel Benjamin Seminario Serna - 20210056G

Introducción a la Compresión de Datos

La compresión de datos sin pérdida es crucial para reducir el tamaño de archivos sin sacrificar información. Permite reconstruir el contenido original, siendo esencial en almacenamiento, transmisión de datos y sistemas embebidos con espacio o ancho de banda limitado.

Algoritmos de Compresión

LZ78 y LZ78_PAR

Desarrollado por Lempel y Ziv en 1978, LZ78 construye un diccionario dinámico. LZ78 PAR es una versión paralela que divide el texto para compresión simultánea, mejorando la velocidad a pesar de la sobrecarga.

LZW

Creado por Terry Welch en 1984, LZW optimiza LZ78 al reutilizar el diccionario continuamente, guardando solo índices de patrones. Fue clave en formatos como GIF y archivos .Z de Unix.

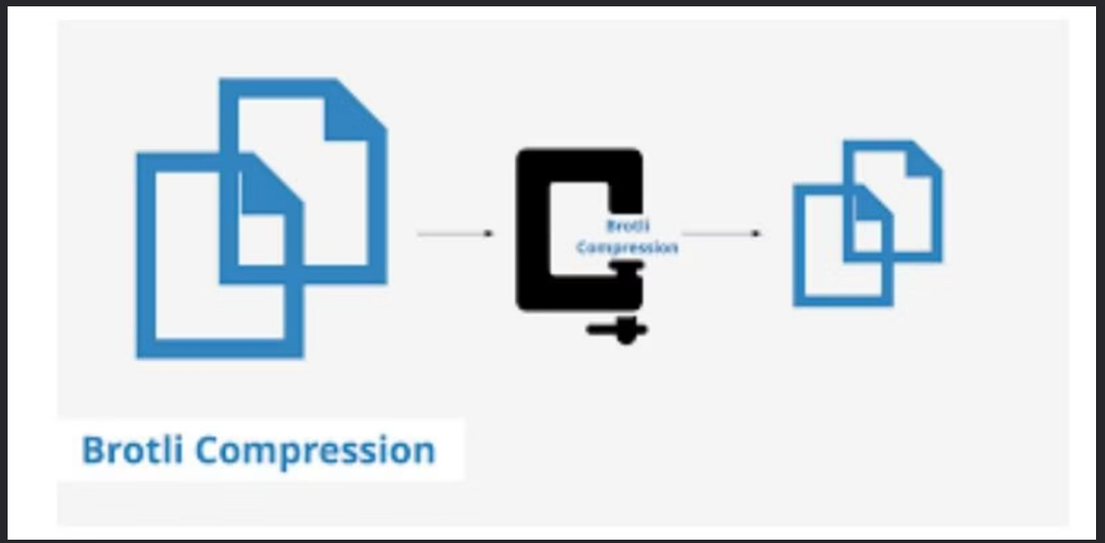
Algoritmos de Compresión

Gzip

Desarrollado por Jean-loup Gailly y Mark Adler en 1992, Gzip usa el algoritmo DEFLATE (LZ77 + Huffman). Es un estándar en sistemas Unix/Linux y en la web (HTTP/1.1) por su eficiencia y velocidad.

Brotli

Lanzado por Google en 2015, Brotli optimiza LZ77 con contextos, Huffman adaptativo y un diccionario preentrenado de 120KB. Ofrece mejores ratios que Gzip, especialmente en la web (HTTP/2 y HTTP/3).



Objetivo del Proyecto

Este proyecto busca evaluar y comparar algoritmos de compresión sin pérdida en textos de cinco idiomas (Quechua, Español, Inglés, Portugués y Ruso).

Ratio de Compresión

Analizar la eficiencia de cada algoritmo en la reducción del tamaño de los archivos.

Tiempo de Ejecución

Medir la velocidad de compresión de cada algoritmo.

Eficiencia Multilingüe

Comparar el rendimiento en diferentes idiomas.

Impacto Tecnológico

Evaluar la influencia del paralelismo y el lenguaje de implementación.

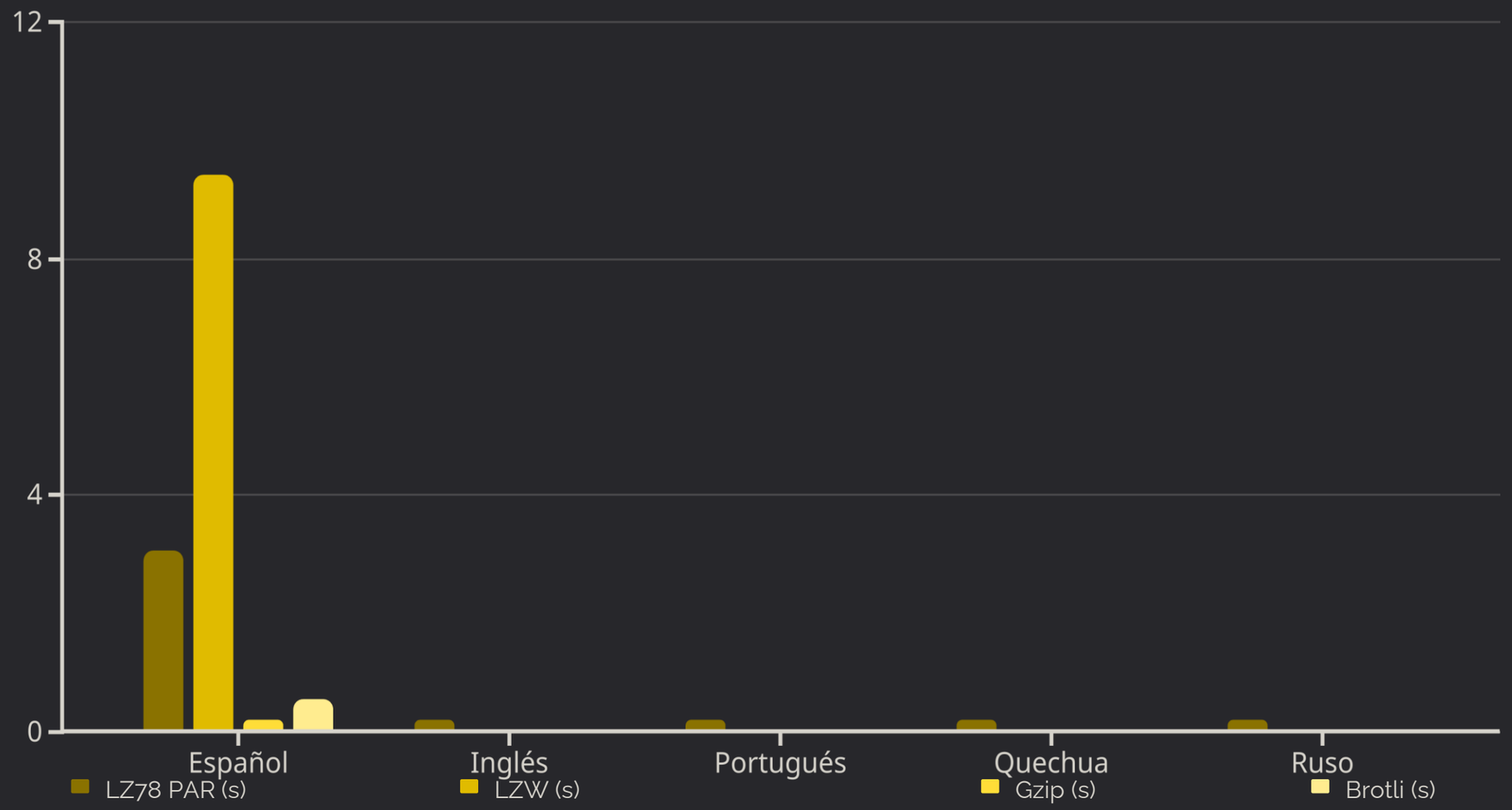
Metodología y Resumen Técnico

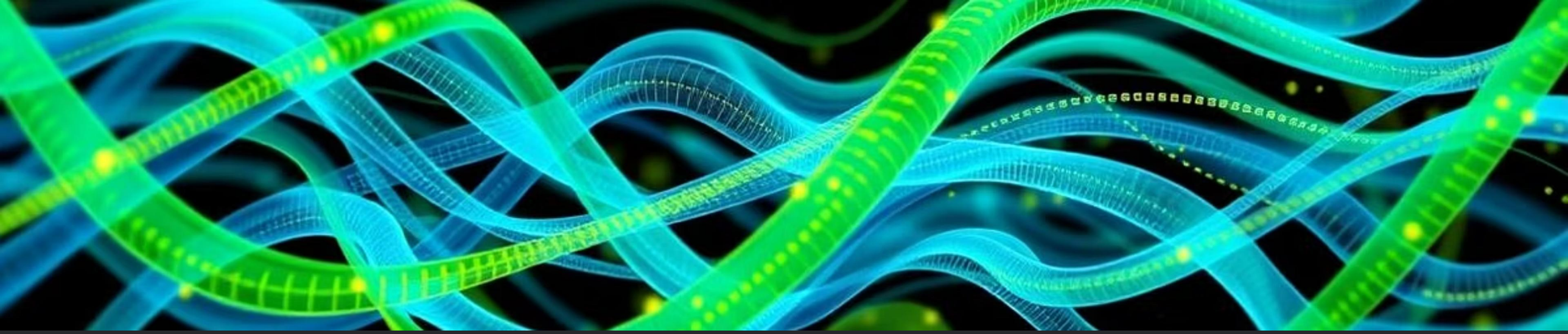
Se utilizaron textos en cinco idiomas, con un corpus más grande para el español (más de 20 millones de caracteres) para resaltar diferencias de rendimiento.

LZ78 PAR	compresores/z78_parallel.py	Python	Sí (multiprocessing)	Diccionarios independientes por fragmento. Buena velocidad con sobrecarga.
LZW	compresores/lzw.py	Python	No	Diccionario secuencial. Sin paralelismo.
Gzip	gzip (binario externo)	Binding en C	No	Combina LZ77 + Huffman. Muy eficiente y rápido.
Brotli	brotli (módulo)	Binding en C++	No	Usa contexto + Huffman adaptativo + diccionario global optimizado.

Resultados e Interpretación

Brotli y Gzip obtuvieron los mejores ratios de compresión. LZ78 PAR fue más rápido que LZW gracias al paralelismo, a pesar de ser Python. LZW fue el más lento por su naturaleza secuencial y lenguaje interpretado.





Conclusión

Brotli y Gzip destacan por su eficiencia algorítmica e implementación en lenguajes de bajo nivel. LZ78 PAR demuestra el valor del paralelismo, incluso en Python. La elección del algoritmo debe considerar el contexto: lenguaje, tipo de datos, arquitectura y requerimientos de tiempo/memoria.