27/10/2025

CRACKING DE CONTRASEÑAS MEDIANTE FUERZA BRUTAS APLICANDO ENFOQUE DyV

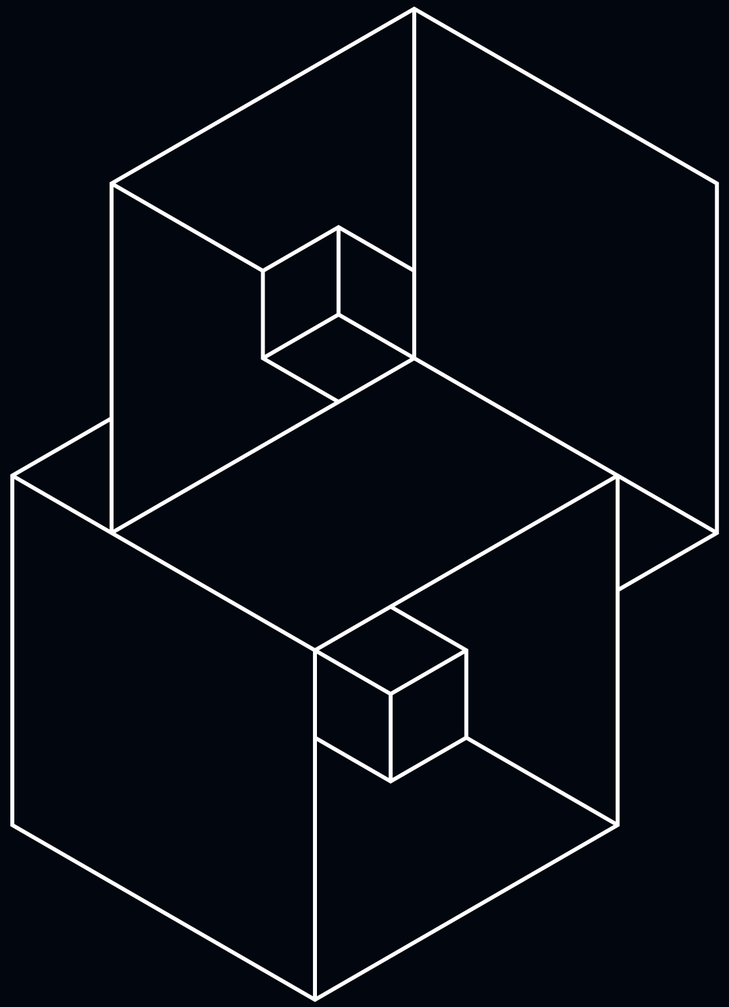
Equipo:	Códigos:
Eduardo Ramos Ochoa	304489918
Andres Santiago Aguirre Macias	220293594
Cesar Emmanuel Gómez Martínez	214843698

Análisis de Algoritmos
Centro Universitario de Ciencias Exactas e Ingenierías (CUCEI)

Python 3.13.9

INDICE

- Introducción
- Funcionamiento del Codigo
- Espacio temporal y espacial
- Comparación de Fuerza bruta VS DyV
- Conclusiones



Introducción

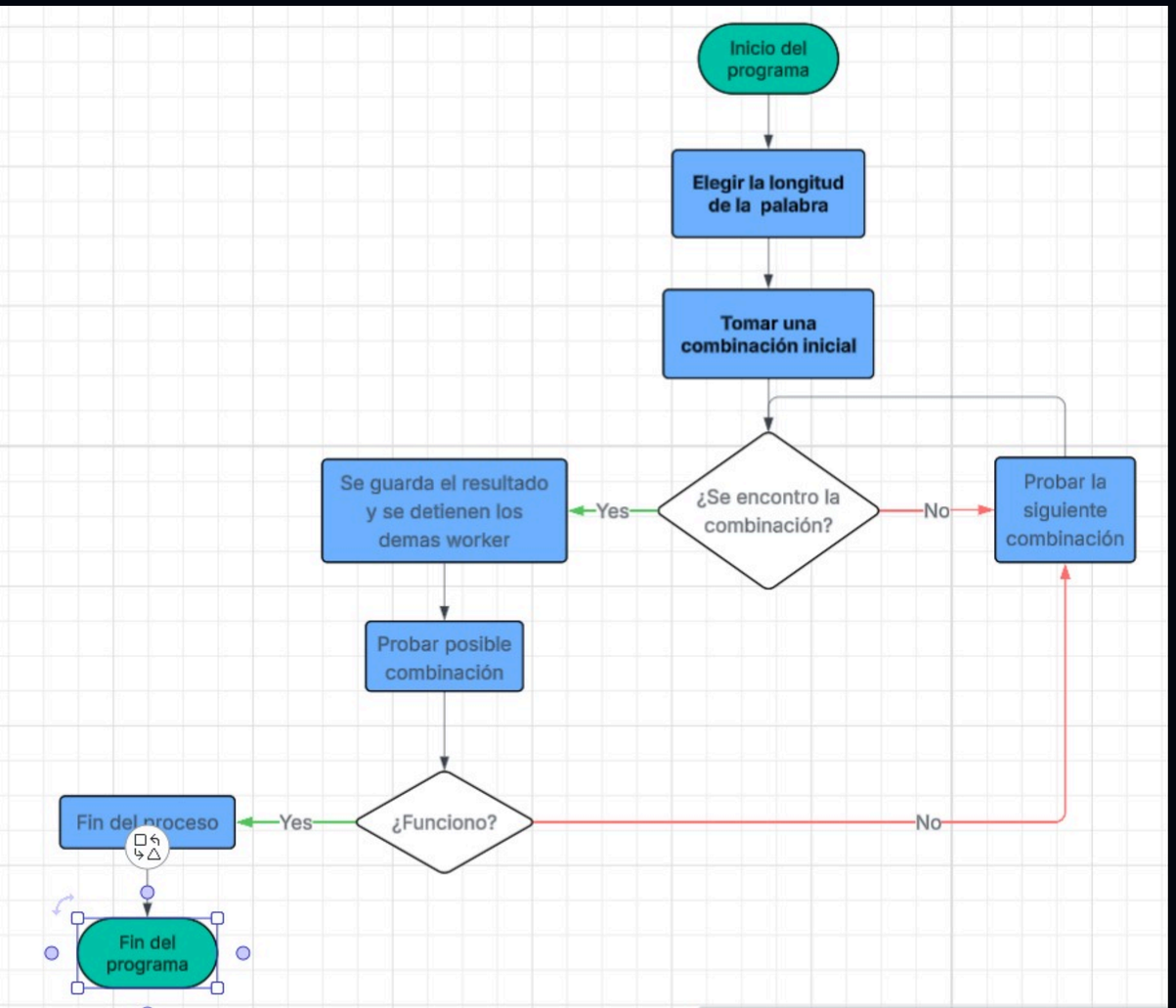
Realizar una comparación en materia de "Cracking de contraseñas" a través de dos diferentes enfoques, los cuales son; Fuerza Bruta y, la aplicación del enfoque Divide y Vencerás al mismo algoritmo

CRACKING DE CONTRASEÑAS MEDIANTE FUERZA BRUTAS APLICANDO ENFOQUE DyV

El enfoque de divide y vencerás, esta basado en la partición: el espacio de búsqueda puede dividirse en subespacios independientes (por ejemplo, por prefijos).

Funcionamiento del Codigo

Diagrama de flujo



pseudocódigo

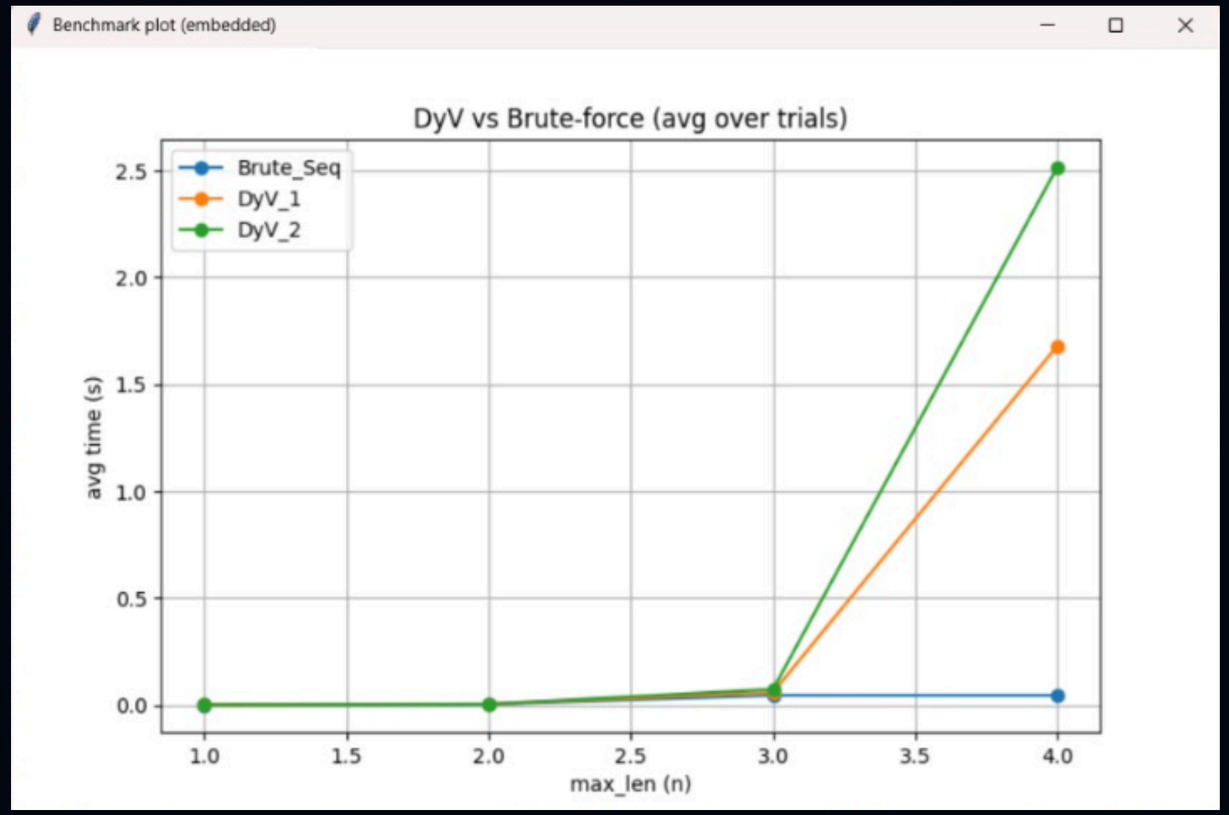
❖ **A CONTINUACIÓN, SE PRESENTA EL "PSEUDOCÓDIGO" PROPUESTO:**

```
FuerzaBruta_DyV(alphabet, max_len, target_hash, p, prefix_len):
    prefixes = all_combinations(alphabet, length=prefix_len)
    groups = repartir_round_robin(prefixes, p)
    shared_result = None
    stop_event = false

    para i en 1..p:
        lanzar worker(i, groups[i])

    worker(i, pref_list):
        para length en prefix_len..max_len:
            para cada pref en pref_list:
                si stop_event: salir
                si len(pref)==length:
                    probar pref
                sino:
                    para suf en product(alphabet, repeat=length-
len(pref)):
                        candidato = pref + suf
                        probar candidato
                        si coincide:
                            shared_result = candidato
                            stop_event = true
                            salir

    esperar a que terminen workers
    retornar shared_result
```

Complejidad temporal

$$O(|\Sigma|^n / p)$$

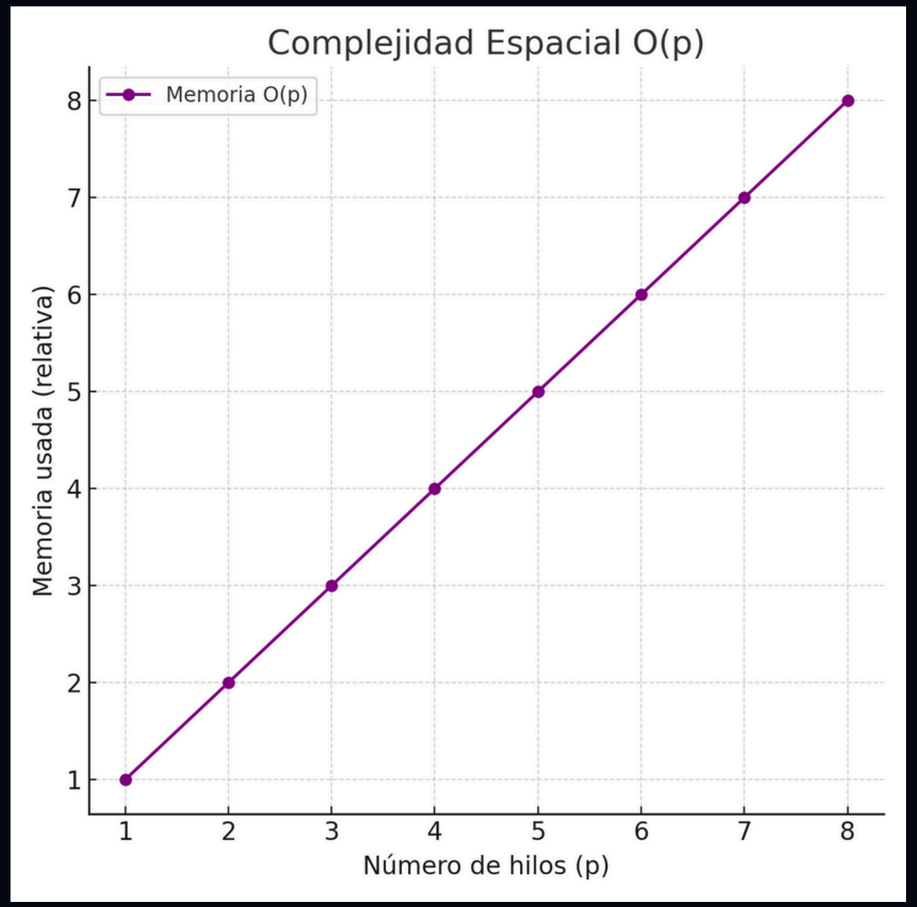
- **|Σ| → cuántos caracteres hay para probar (alfabeto).**
- **n → qué tan larga puede ser la contraseña.**
- **p → cuántos hilos o núcleos del procesador están trabajando al mismo tiempo.**
- **O(...) → cuánto crece el tiempo de ejecución conforme el problema se hace más grande.**



Complejidad Espacial

O(p)

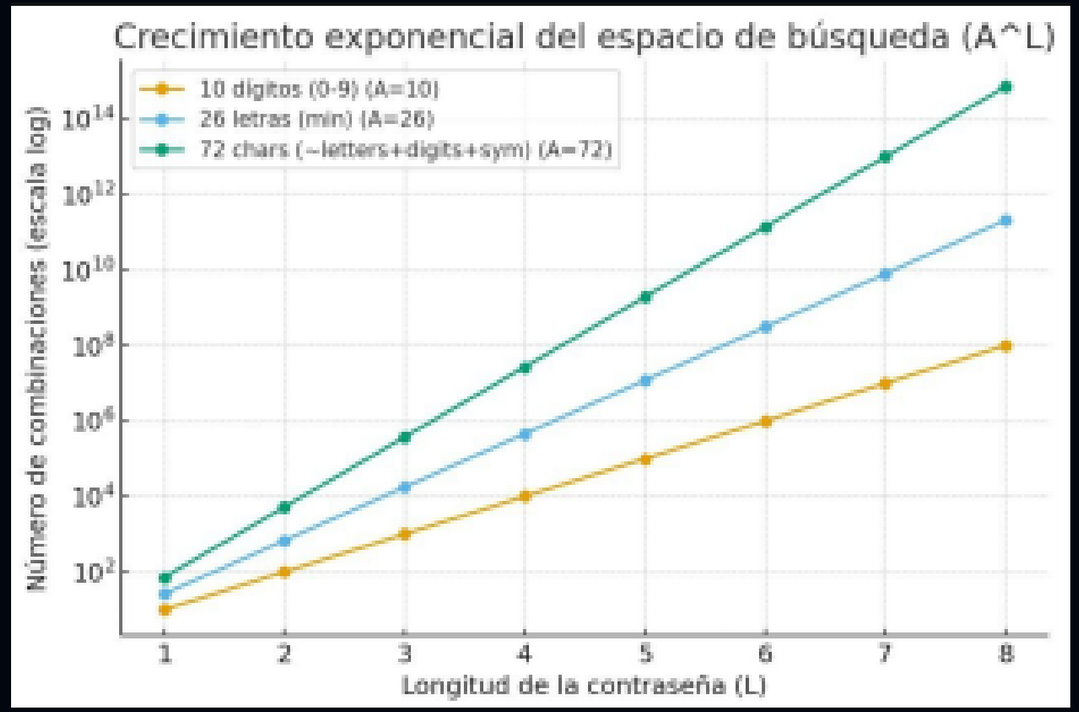
- Cada hilo usa poca memoria
- Constante por hilo (prefijo, hash)
- No depende del tamaño del espacio
- Memoria total crece linealmente



Comparación de Fuerza bruta VS DyV

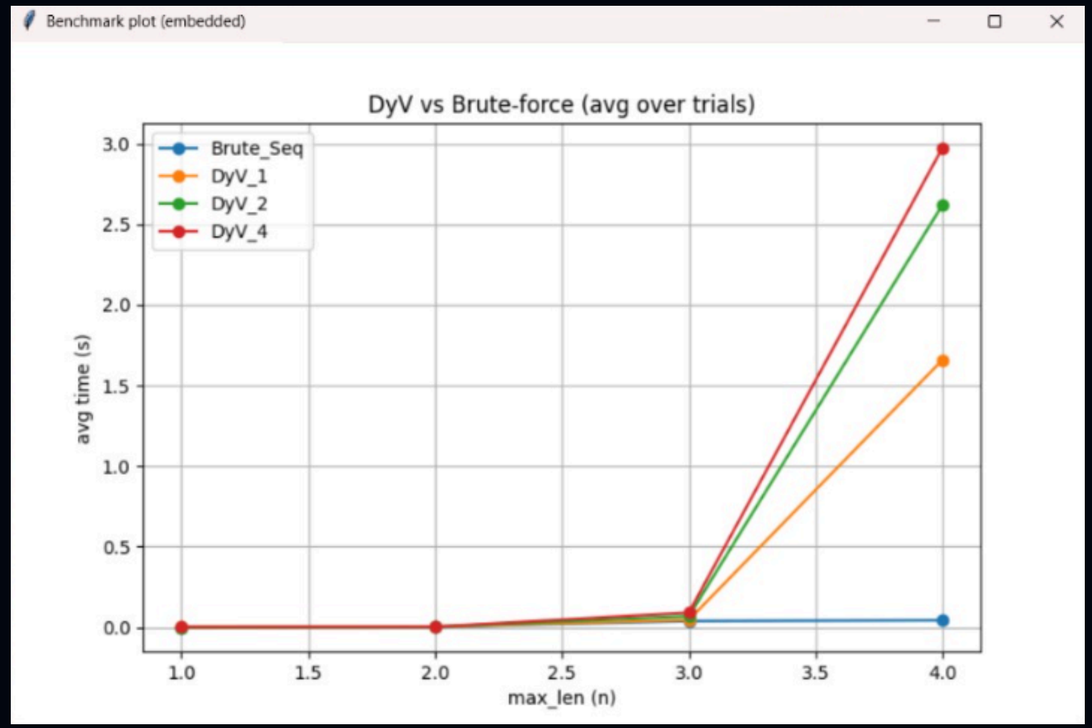
Fuerza bruta

- Probar todas las combinaciones posibles una por una, en orden.
- No utiliza paralelismo; todo secuencial.
- Lento si el espacio es grande



DyV

- Dividir el espacio de búsqueda en subespacios (prefijos) y asignarlos a varios workers/hilos para explorar en paralelo.
- Se ejecuta en múltiples hilos o procesos.



Conclusiones

El análisis de algoritmos mediante la complejidad temporal y espacial permite observar cómo se comportan distintos métodos frente a un mismo problema. En este caso, el algoritmo de fuerza bruta resultó más eficiente que el enfoque de Divide y Vencerás, mostrando menor tiempo de ejecución y consumo de recursos para los tamaños de problema evaluados, aunque, la eficiencia depende del tamaño de entrada "N" esto es, para volúmenes superiores o mayores, DyV es mejor.

Bibliografias

- Fortinet. Brute Force Attack – Definition and Explanation. Fortinet. https://www.fortinet.com/lat/resources/cyberglossary/brute-force-attack

- Kaspersky. ¿Qué es un ataque de fuerza bruta? Kaspersky. https://www.kaspersky.es/resource-center/definitions/brute-force-attack

- freeCodeCamp. Significado del algoritmo Divide y Vencerás. freeCodeCamp. https://www.freecodecamp.org/espanol/news/significado-del-algoritmo-divide-y-venceras/

- LinkedIn. Ventajas y desventajas del uso de fuerza bruta y alternativas. LinkedIn. Agregar un subtítulo

Contactos

Correos institucionales

GitHub

Documentación