Algoritmos

Trabajo Práctico N1- 2024s1

Alumno: Martín Castello

Introducción

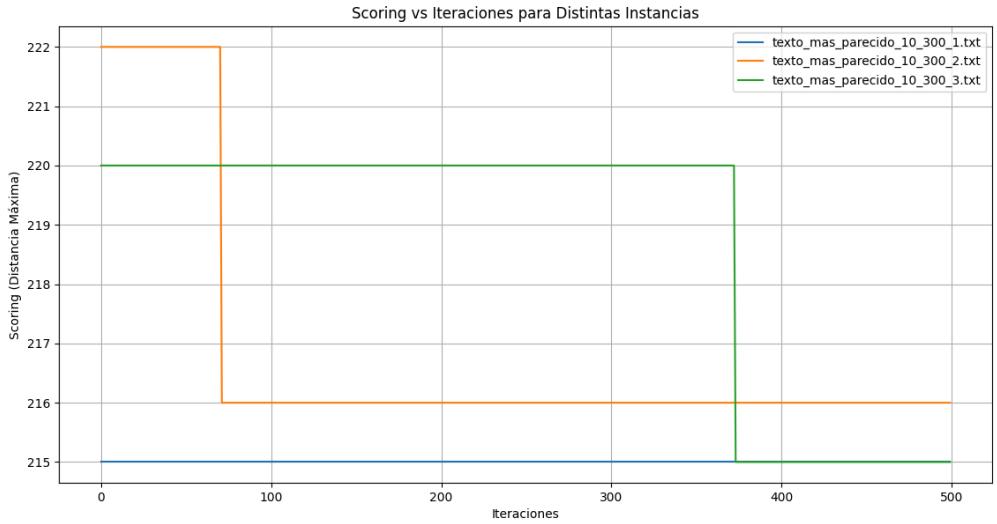
1. Proponer un algoritmo goloso para el problema del texto más parecido.
2. Aleatorizar el algoritmo anterior.
3. Proponer un algoritmo de búsqueda local para el problema del texto más parecido .
4. Variar parámetros y la estrategia del algoritmo de búsqueda local que optimicen el funcionamiento del mismo.
5. Construir un algoritmo GRASP para el problema del texto más parecido. La entrada de su algoritmo será un archivo con una instancia del problema del texto más parecido (un texto por línea) y la salida deberá ser un archivo de texto plano con una secuencia en la primera línea y su distancia máxima en la siguiente)
6. Presentar un gráfico de scoring contra la cantidad de iteraciones para baterías de distintas instancias, que permita decidir una cantidad de iteraciones que ayude a encontrar un valor cercano al óptimo sin desperdiciar tiempo de cómputo.

Texto más parecido 300

Texto más parecido 10_300

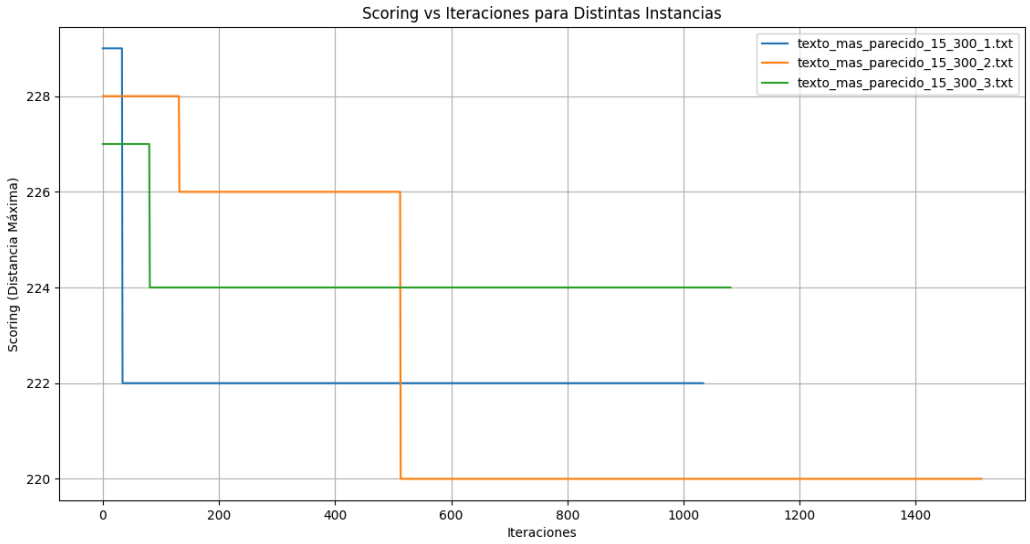
En el siguiente gráfico podemos ver la ejecución de 3000 iteraciones para cada archivo de texto_mas_parecido_10_300:

Como se puede observar, a partir de la iteración 250 se mantiene el scoring constante (sin bajar). Por ende, podemos ejecutar nuevamente pero sin superar las 500 iteraciones y el resultado fue:



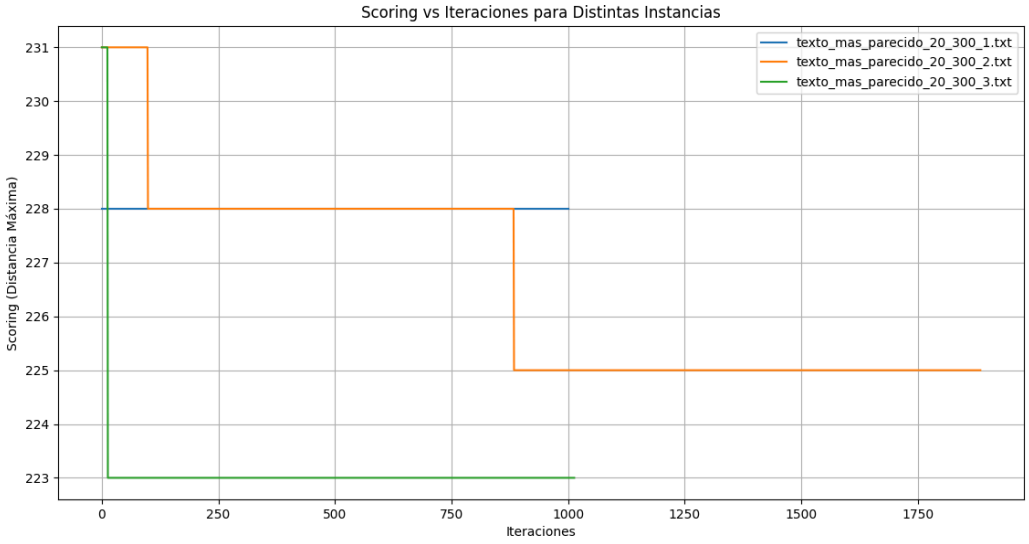
Texto más parecido 15_300

En el siguiente gráfico, se presenta la implementación de un algoritmo con un mecanismo de control de corte basado en el número de iteraciones. Este mecanismo garantiza que el algoritmo no seguirá ejecutándose indefinidamente si no encuentra una mejora en el scoring después de un número considerable de intentos, en este caso 1000:



Texto más parecido 20_300

En el siguiente gráfico también se puede observar que llegado a mil iteraciones, donde se mantiene constante el scoring, corta la ejecución:

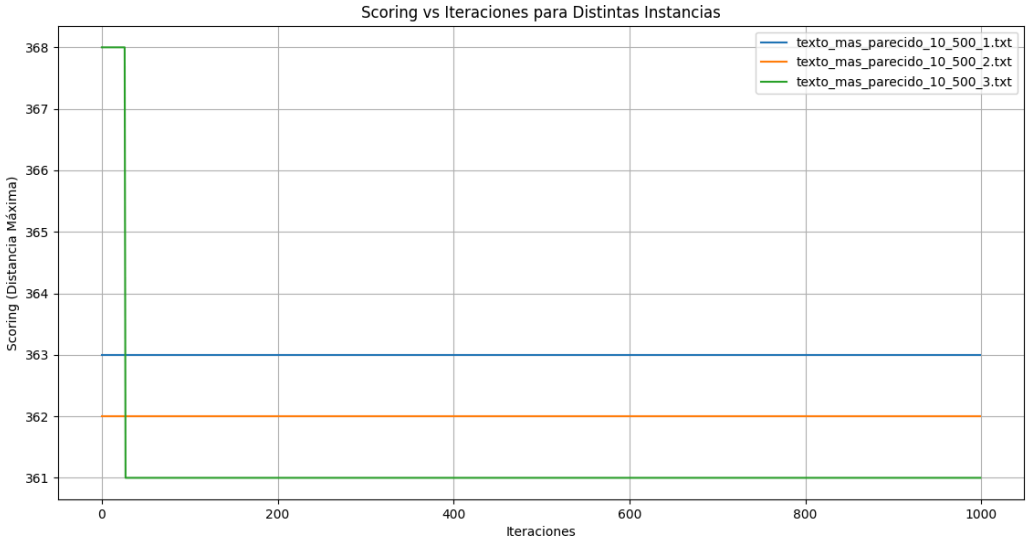


Texto más parecido 500

Texto más parecido 10_500

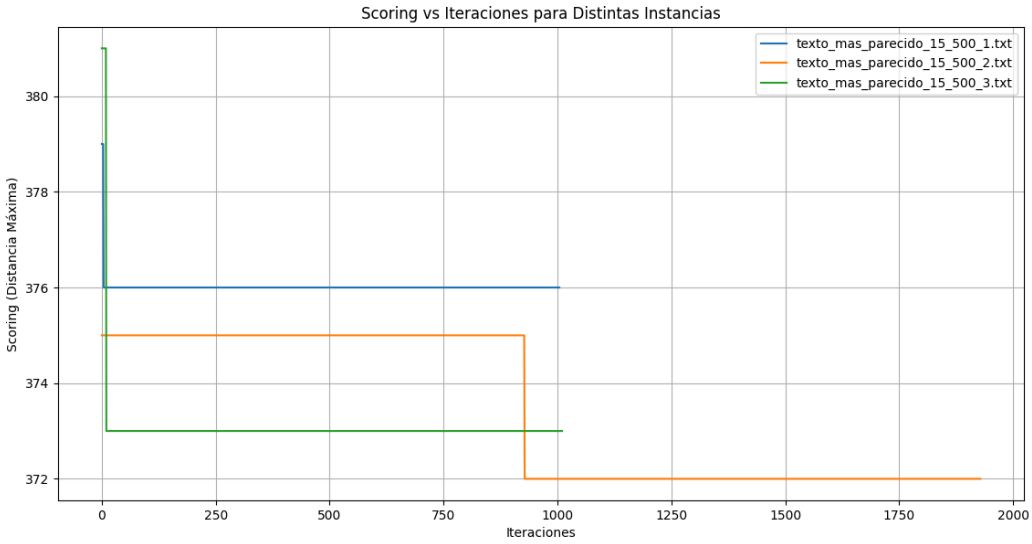
En el siguiente gráfico podemos ver la ejecución de 3000 iteraciones para cada archivo de texto_mas_parecido_10_500:

Como se puede observar, a partir de la iteración 750 se mantiene el scoring constante (sin bajar). Por ende, podemos ejecutar nuevamente pero sin superar las 1000 iteraciones y el resultado fue:



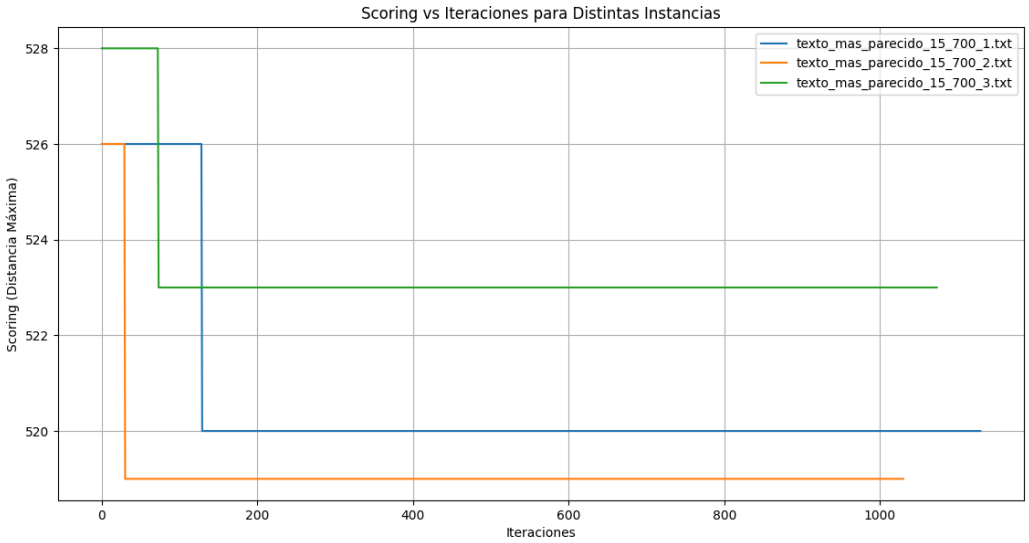
Texto más parecido 15_500

En el siguiente gráfico, se presenta la implementación de un algoritmo con un mecanismo de control de corte basado en el número de iteraciones. Este mecanismo garantiza que el algoritmo no seguirá ejecutándose indefinidamente si no encuentra una mejora en el scoring después de un número considerable de intentos, en este caso 1000:



Texto más parecido 20_500

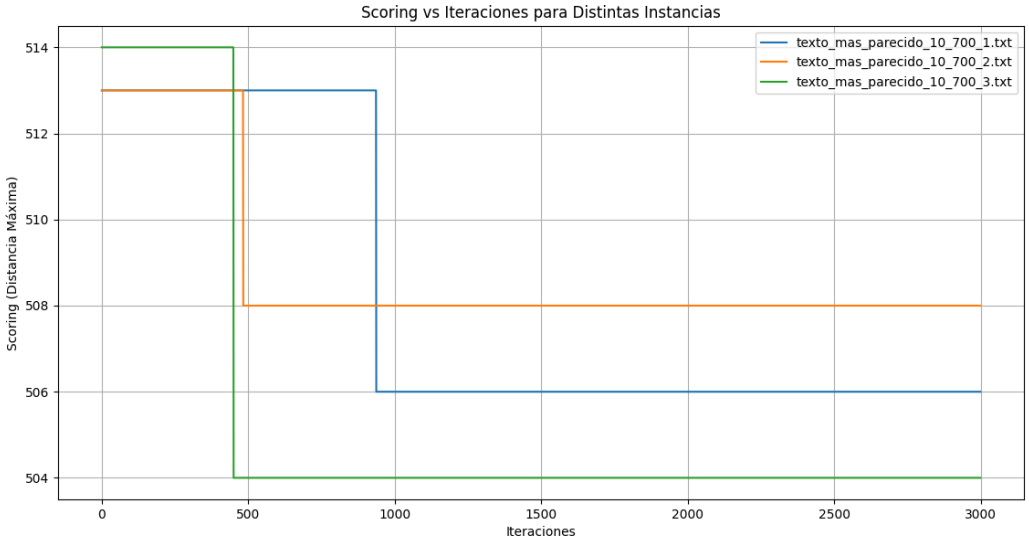
En el siguiente gráfico también se puede observar que llegado a mil iteraciones, donde se mantiene constante el scoring, corta la ejecución:



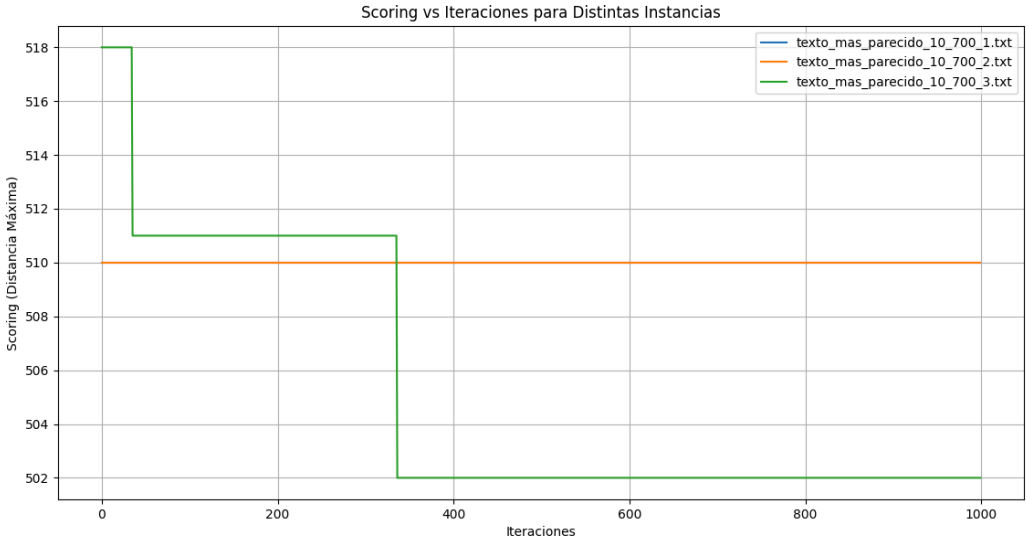
Texto más parecido 700

Texto más parecido 10_700

En el siguiente gráfico podemos ver la ejecución de 3000 iteraciones para cada archivo de texto_mas_parecido_10_700:

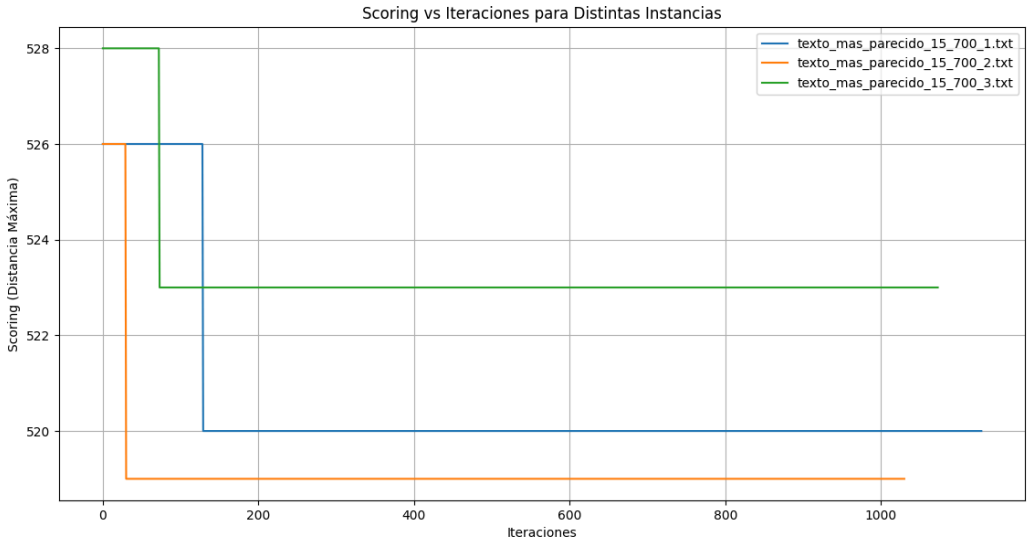


Como se puede observar, a partir de las 1000 iteraciones se mantiene el scoring constante (sin bajar). Por ende, podemos ejecutar nuevamente pero sin superar las 1000 iteraciones y el resultado fue:



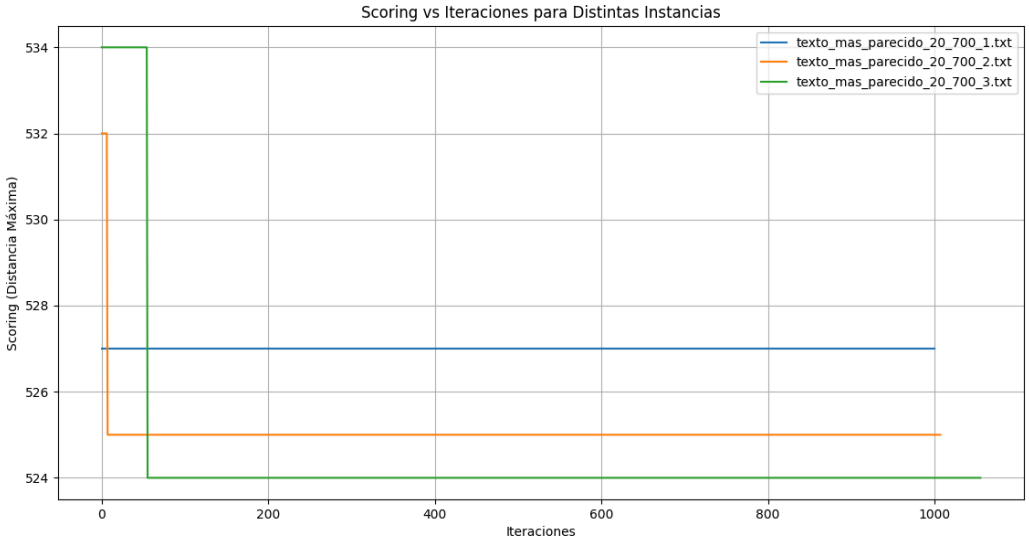
Texto más parecido 15_700

En el siguiente gráfico, se presenta la implementación de un algoritmo con un mecanismo de control de corte basado en el número de iteraciones. Este mecanismo garantiza que el algoritmo no seguirá ejecutándose indefinidamente si no encuentra una mejora en el scoring después de un número considerable de intentos, en este caso 1000:



Texto más parecido 20_700

En el siguiente gráfico también se puede observar que llegado a mil iteraciones, donde se mantiene constante el scoring, corta la ejecución:



Conclusión

Iteraciones vs Entrada

En el siguiente gráfico podemos ver la curva de la función de las iteraciones en base a cada batería de instancias de archivos donde se puede ver que se dio que a más cantidad de entrada no se necesita tantas iteraciones, todo lo contrario a lo que se esperaba ya que como se explicó anteriormente encontró no se encontró un scoring mejor luego de las primeras

iteraciones: