

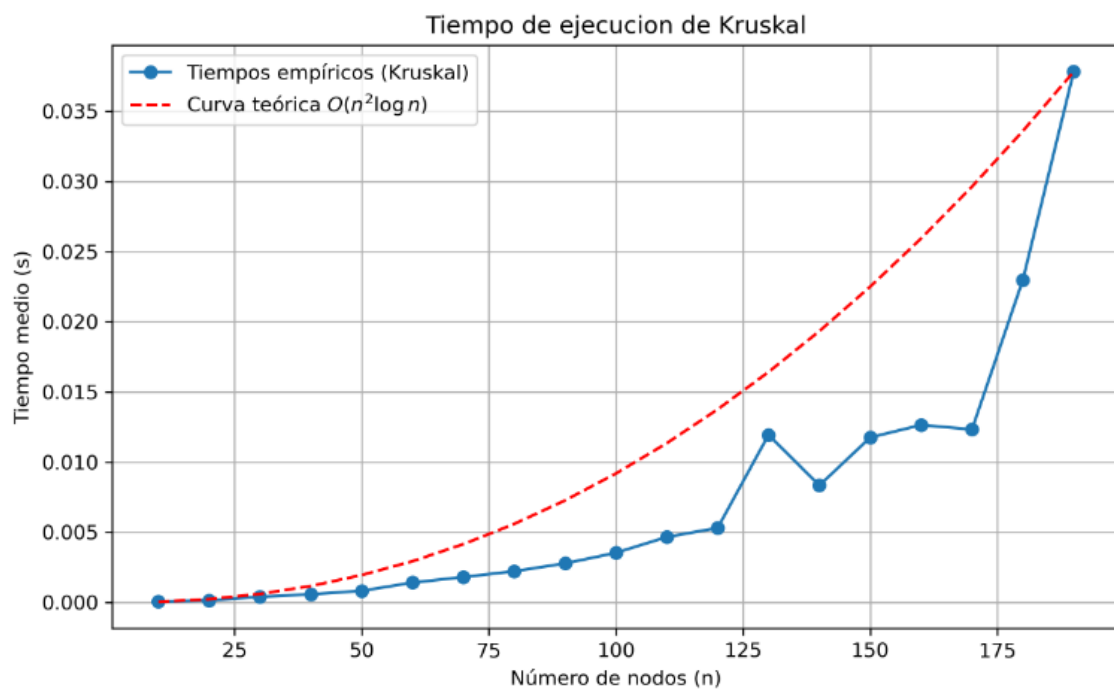
En esta memoria se va a explicar los resultados obtenidos en esta práctica.

## II-B

### Coste de Kruskal

Teóricamente su coste es de  $O(E \cdot \log(E))$ , siendo  $E$  las aristas, al tratarse de un grafo completo el número máximo de aristas es  $E = [n(n-1)]/2$ ,  $O(n^2)$  y viendo  $n^2$  para  $O(E)$ , se tiene que  $n^2 \log(n)$ .

Tras varias ejecuciones se ha comprobado que la recta empírica queda por debajo de la curva teórica por lo que la función es mas eficiente de lo esperado.



## III-B

1-El algoritmo greedy-tsp tiene una complejidad de  $O(n^2)$  ya que examina todos los nodos no visitados (el bucle for) por cada n iteraciones (bucle while) y como repeated\_greedy\_tsp llama la función n veces se deduce que tiene una complejidad de  $O(n^3)$ .

El algoritmo de exhaustive\_tsp genera  $n-1$  permutaciones ya que se inicia en el nodo 0 y como evalúa todas las rutas posibles de  $n$  aristas ya tendría una complejidad de  $O(n!)$ .

2-El algoritmo del vecino cercano no siempre es óptimo porque al buscar el nodo más cercano puede existir el caso de que un nodo tenga una distancia larga el algoritmo ignoraría ese recorrido teniendo una ruta más corta. Por ejemplo:

	A	B	C	D
A	-	2	9	10
B	2	-	6	4
C	9	6	-	3
D	10	4	3	-

Según el algoritmo la ruta de greedy, partiendo de A sería  $2 + 4 + 3 + 9 = 18$  A  $\rightarrow$  B  $\rightarrow$  D  $\rightarrow$  C  $\rightarrow$  A, pero lo óptimo sería A  $\rightarrow$  B  $\rightarrow$  C  $\rightarrow$  D  $\rightarrow$  A, de longitud  $2 + 6 + 3 + 10 = 21$ .