Informe de modificaciones

Se reentrega únicamente la sección referente al algoritmo exacto. Las modificaciones en el informe están marcadas en negrita, mientras que todo lo que se mantuvo igual está en color gris.

Implementación del Algoritmo

Se implementó una nueva poda y se agregó toda la explicación correspondiente, incluyendo la justificación de por qué la poda es correcta. Se agregó al pseudocódigo la utilización de dicha poda y se comentó esta modificación en el pseudocódigo. Se agregó una explicación de por qué no se incluyó el pseudocódigo de la función Max. Se agregó el pseudocódigo de la función *se_conecta_con_clique* y un breve análisis de su complejidad. También se amplió la explicación sobre el pseudocódigo haciendo énfasis en la estructura utilizada para representar una solución y la forma en que eso afecta la complejidad de modificar una solución.

Orden de complejidad

Se corrigió la complejidad del algoritmo. También se modificó la oración que aseguraba que la complejidad lineal de cada iteración no modificaba la complejidad de la cantidad de iteraciones (de lo cual habíamos deducido erróneamente que n,$2^n \in \mathrm{O}(2^n)$). Se agregaron también las referencia a la complejidad de la función *se_conecta_con_clique* y a la complejidad de modificar una solución, ambas basándose en las estructuras utilizadas para representar un grafo.

Experimentación

Se rehizo el gráfico de performance para comparar las dos versiones anteriores (con poda y sin poda) con la nueva versión del algoritmo (2 podas). En la primera entrega se ejecutó la versión con poda hasta $n = 50$. Para poder comparar todas las versiones en los mismos tamaños de entrada, esta vez se recortó a 30 el tamaño máximo de entrada.

Se realizó un segundo test para comparar las tres versiones en el peor caso, es decir, con grafos completos. En este segundo test, todas las versiones tienen altos tiempos de ejecución, por lo que no se pudo ejecutar con más de 30 nodos. Con el fín de comparar ambos test, se decidió que 30 sea el límite para ambos (otra razón para reducir el máximo tamaño de entrada en el primer test). Para este segundo test se implementó una nueva función de la clase grafo para generar grafos aleatorios. Se graficó y analizó el resultado del segundo test.