

Capítulo 4 Experiencia computacional

Se evalúan 4 algoritmos de generación de columnas para el problema de corte:

- Algoritmo 1 (ver Anexo 1.1).- Este programa es el algoritmo mostrado en la sección 3.2 el cual no contiene ninguna heurística para mejorar la eficiencia.
- Algoritmo 2 (ver Anexo 1.2).- Programa con el algoritmo mostrado en la sección 3.2, que incluye la heurística mostrada en la sección 3.2.1.2 para encontrar soluciones factibles para el problema de la mochila que debe resolverse en cada iteración del método de generación de columnas.
- Algoritmo 3 (ver Anexo 1.3).- Programa con el algoritmo mostrado en la sección 3.2, que incluye la heurística mostrada en la sección 3.2.1.1 para generar una solución inicial factible que incluya patrones de corte combinados (patrones que incluyen cortes con distintos anchos).
- Algoritmo 4 (ver Anexo 1.4).- Programa con el algoritmo mostrado en la sección 3.2, que incluye la heurística mostrada en la sección 3.2.1.1 para generar una solución inicial factible que incluya patrones de corte combinados y la heurística mostrada en la sección 3.2.1.2, para encontrar soluciones factibles para el problema de la mochila que debe resolverse en cada iteración del método de generación de columnas para identificar patrones de corte adicionales y además, la heurística.

Se generó de manera aleatoria, un conjunto con 25 instancias de prueba con diferentes números de anchos de la siguiente manera:

- 5 instancias de prueba con 5 anchos.
- 5 instancias de prueba con 20 anchos.
- 5 instancias de prueba con 30 anchos.
- 5 instancias de prueba con 40 anchos.
- 5 instancias de prueba con 50 anchos.

El objetivo de las pruebas computacionales es comparar la eficacia y la eficiencia de los distintos algoritmos a medida que aumenta en número de anchos, y por tanto la complejidad para resolver el problema.

Para comparar la eficiencia de los distintos algoritmos de generación de columnas se reporta: 1) el tiempo de ejecución y 2) el número de iteraciones. Todas las pruebas se ejecutaron en una computadora con un procesador Intel Core Duo de 2.00 GHz y 3.00 GB de memoria en RAM. El lenguaje de programación utilizado fue Mosel XPRESS.

En la siguiente sección se muestran los resultados obtenidos.

4.1 Resultados

En esta sección se muestran los resultados obtenidos para las instancias del problema generadas para evaluar el comportamiento de los distintos algoritmos y que van desde 5 hasta 50 anchos diferentes. También se muestra una evaluación tanto de la eficacia (calidad de las soluciones) como de la eficiencia (esfuerzo computacional) de los distintos algoritmos de generación de columnas.

La tabla que se muestra a continuación muestra los resultados obtenidos (tiempo de ejecución en segundos de CPU y número de iteraciones), de los problemas que contienen 5 anchos (ver Anexo 2).

Tabla 1.- Tiempo de Ejecución y Número de Iteraciones para instancias con 5 anchos (Elaboración Propia).

	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones
1	0.062	2	0.042	2	0.076	3	0.03	1
2	0.193	7	0.175	7	0.181	6	0.176	6
3	0.079	3	0.036	3	0.063	2	0.046	2
4	0.102	4	0.083	4	0.031	0	0.031	0
5	0.078	2	0.037	2	0.032	0	0.032	0
Totales	0.514	18	0.373	18	0.383	11	0.315	9

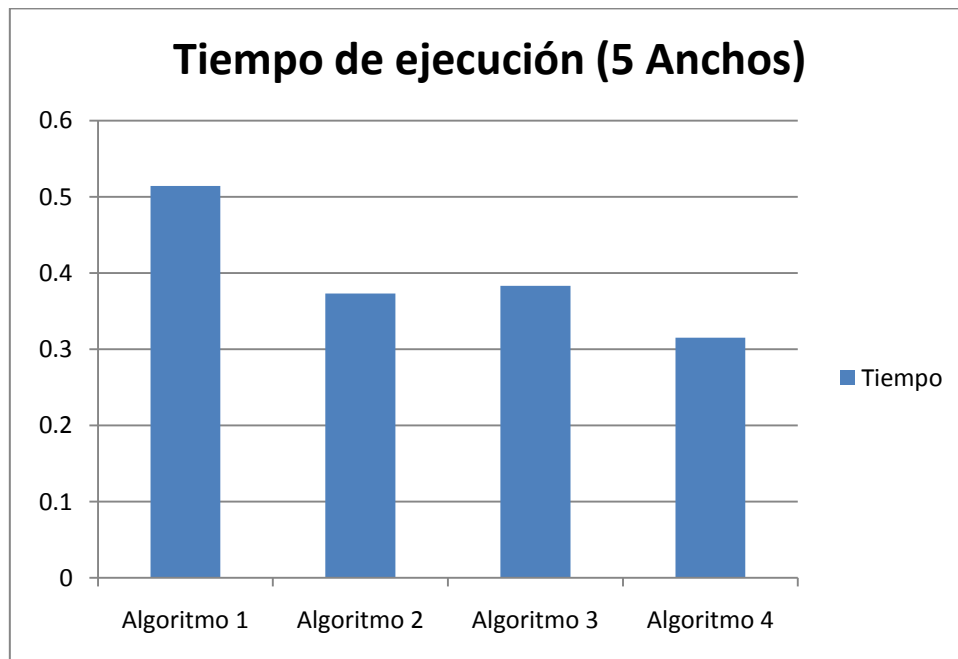


Ilustración 1.- Comparación entre los tiempos de ejecución para instancias con 5 anchos (Elaboración Propia).

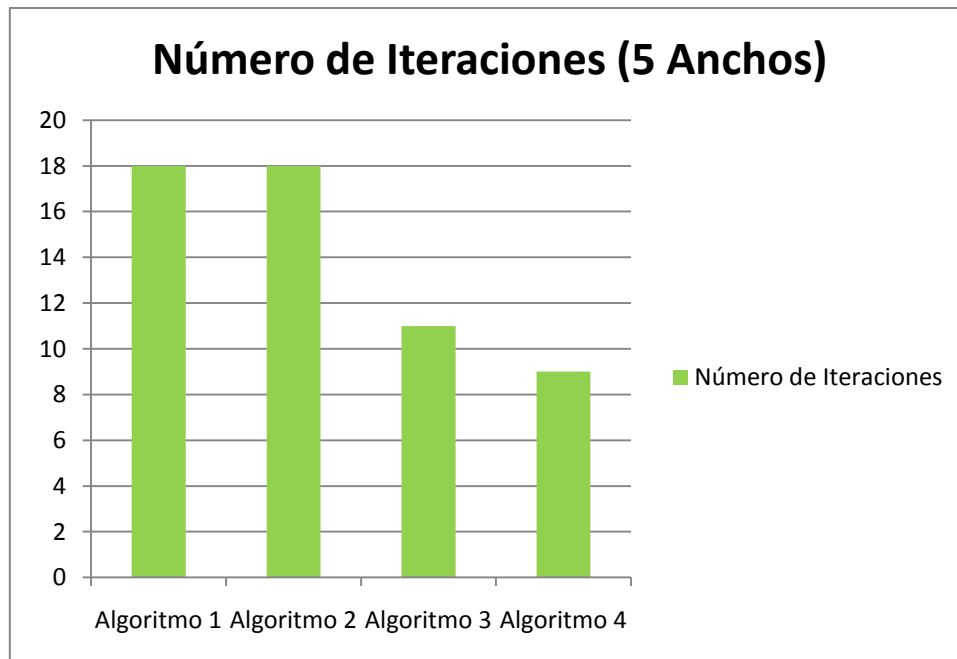


Ilustración 2.- Comparación entre el número de iteraciones para instancias con 5 anchos (Elaboración Propia).

En la Tabla 3 y en las Ilustraciones 8 y 9 se puede observar una disminución tanto en el número de iteraciones como en el tiempo de ejecución, para instancias pequeñas del problema, cuando se incluyen las dos heurísticas. Entre el “Algoritmo 1” y el “Algoritmo 4” se observa una disminución de 39% en el tiempo de ejecución y del 50% en el número de iteraciones.

Con respecto a la calidad de las soluciones obtenidas con cada uno de los algoritmos de generación de columnas, en la Tabla 4 y en la Ilustración 10 podemos observar que no hay diferencia entre el *gap* relativo (brecha de dualidad) de los diferentes algoritmos. Por lo que en instancias pequeñas del problema todos los Programas generan el mismo resultado. De la misma manera podemos observar que el Gap Relativo es muy pequeño (menor al 0.6% en el peor de los casos), lo que nos sugiere que los distintos algoritmos generan soluciones óptimas o soluciones factibles cercanas a las soluciones óptimas.

Tabla 2.- Cota Inferior, Cota Superior y Gap Relativo para instancias de prueba con 5 anchos (Elaboración Propia).

	Cota Inferior	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
		Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	538.500	539	0.000%	539	0.000%	539	0.000%	539	0.000%
2	407.832	408	0.000%	408	0.000%	408	0.000%	408	0.000%
3	841.000	841	0.000%	841	0.000%	841	0.000%	841	0.000%
4	596.800	597	0.000%	597	0.000%	597	0.000%	597	0.000%
5	368.000	370	0.543%	370	0.543%	370	0.543%	370	0.543%

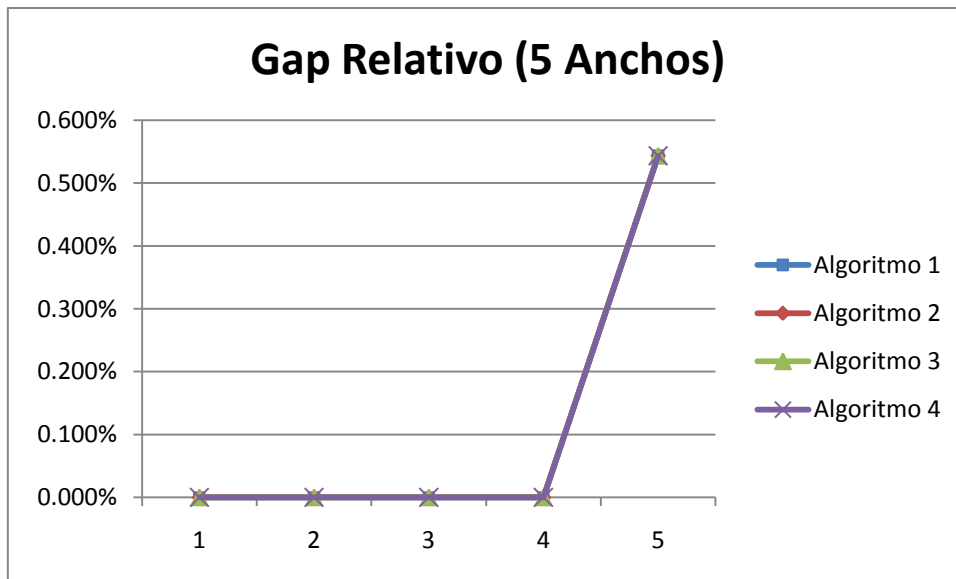


Ilustración 3.- Comparación del Gap Relativo entre los 4 Algoritmos (Elaboración Propia).

La tabla que se muestra a continuación resume los resultados obtenidos (tiempo de ejecución en segundos de CPU y número de iteraciones) para las instancias con 20 anchos diferentes (ver Anexo 2).

Tabla 3.-Tiempo de Ejecución y Número de Iteraciones para instancias con 20 anchos (Elaboración Propia).

	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones
1	1.794	51	1.092	61	1.404	41	1.17	55
2	1.061	34	0.733	45	0.967	28	0.624	33
3	1.732	41	1.295	51	1.748	40	1.7	44
4	1.467	41	0.983	50	1.108	28	0.982	33
5	1.404	43	0.998	54	1.248	36	1.03	48
Totales	7.458	210	5.101	261	6.475	173	5.506	213

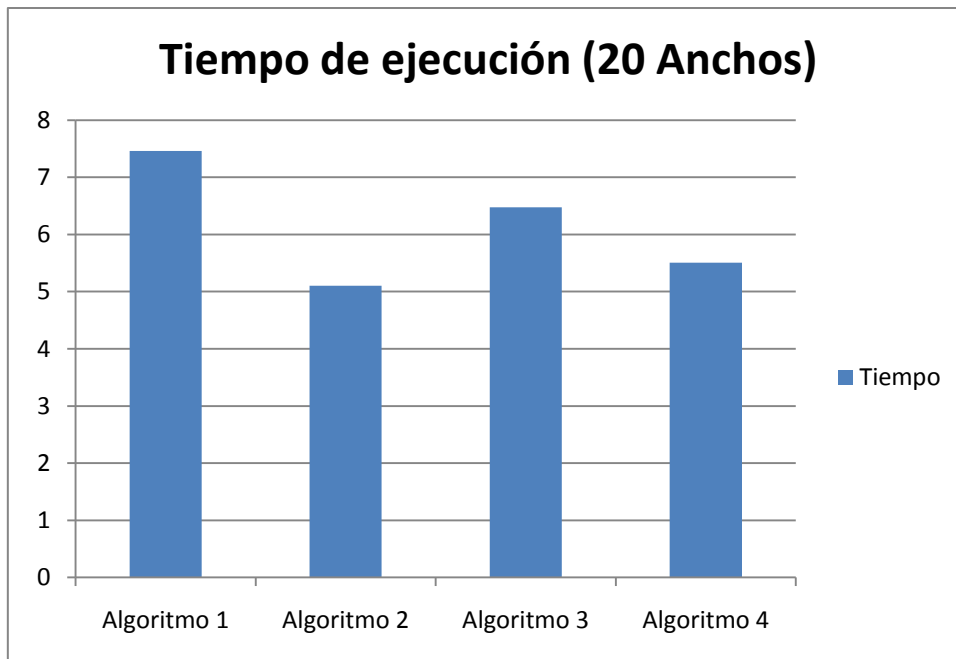


Ilustración 4.-Comparación entre los tiempos de ejecución para instancias con 20 anchos (Elaboración Propia).

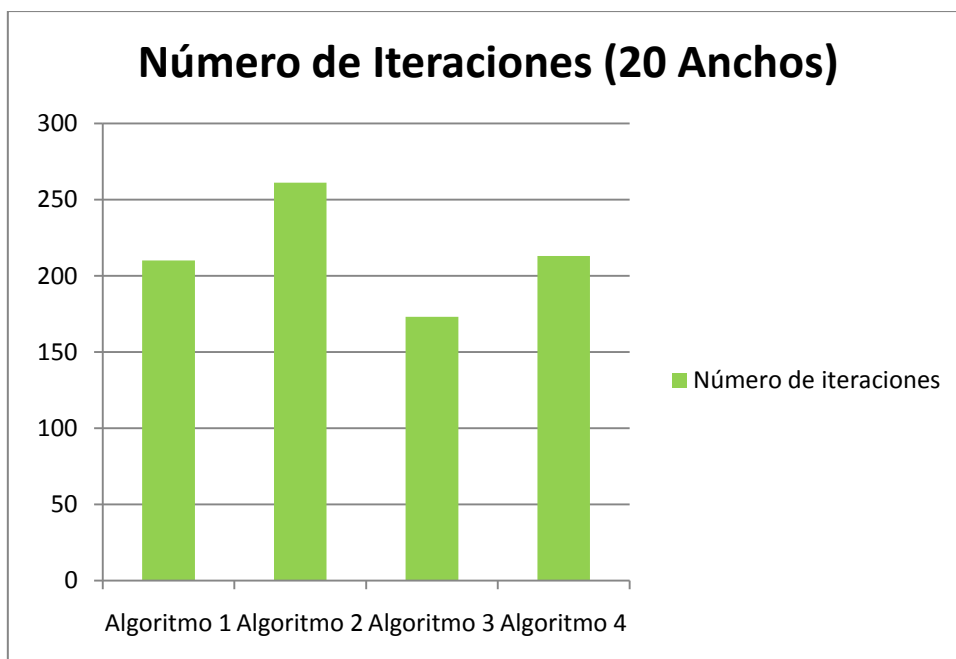


Ilustración 5.- Comparación entre el número de iteraciones para instancias con 20 anchos (Elaboración Propia).

Como se puede observar en la Tabla 5 y en las Ilustraciones 11 y 12, el algoritmo con el peor desempeño con respecto al tiempo de ejecución fue el “Algoritmo 1”, y el que tuvo el mejor desempeño con respecto al tiempo de ejecución fue el “Algoritmo 2” siendo casi 32% más rápido que el “Algoritmo 1”. Una posible interpretación es la siguiente. El “Algoritmo 2” fue más rápido que el “Algoritmo 4” por las siguientes razones. En la Tabla 1, podemos observar que para dos instancias el “Algoritmo 4”, encontró la solución óptima de la relajación lineal del problema con los patrones generados con la heurística y por tanto, no tuvo que hacer ninguna iteración.

Esto es algo que es común que suceda en instancias pequeñas del problema, sin embargo mientras el tamaño va creciendo, es más raro que suceda. Por otro lado también se observa que aunque el número de iteraciones del “Algoritmo 2” es mayor al del “Algoritmo 4”, su tiempo de ejecución sigue siendo menor. Esto nos sugiere que el proceso de obtener patrones iniciales, consume tiempo, y para instancias del problema de tamaño medio se ocupa tanto tiempo en generar la matriz básica inicial que incluso si se resuelve el problema de la mochila a través de la heurística no alcanza a recuperar el tiempo perdido. Si comparamos el número de iteraciones para las instancias de este grupo, podemos observar una tendencia que se mostrará a partir de ahora. El algoritmo que requiere menor cantidad de iteraciones es el “Algoritmo 3” (teniendo casi 19% menos iteraciones que el Algoritmo 1), y el que requiere mayor número de iteraciones es el “Algoritmo 2”. Esto sucede, porque al resolver el problema de la mochila a través de la heurística, no necesariamente se encuentra el patrón óptimo de corte, por lo que genera una mayor cantidad de patrones de corte, sin embargo, a medida que incrementa el número de anchos la heurística del problema de la mochila es capaz de generar los patrones de corte rápidamente.

En la Tabla 6 y en la Ilustración 13 se puede observar el *gap* relativo para cada uno de los cuatro algoritmos. Al igual que para las instancias con 5 anchos, no hay diferencias con respecto al *gap* relativo para los distintos algoritmos. En el caso de las instancias con 20 anchos, las soluciones obtenidas con los cuatro algoritmos son las soluciones óptimas del problema de corte.

Tabla 4.-Cota Inferior, Cota Superior y Gap Relativo para instancias de prueba con 20 anchos (Elaboración Propia).

	Cota Inferior	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
		Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	1812.30	1813	0.000%	1813	0.000%	1813	0.000%	1813	0.000%
2	1901.54	1902	0.000%	1902	0.000%	1902	0.000%	1902	0.000%
3	1872.49	1873	0.000%	1873	0.000%	1873	0.000%	1873	0.000%
4	1726.03	1727	0.000%	1727	0.000%	1727	0.000%	1727	0.000%
5	1888.31	1889	0.000%	1889	0.000%	1889	0.000%	1889	0.000%

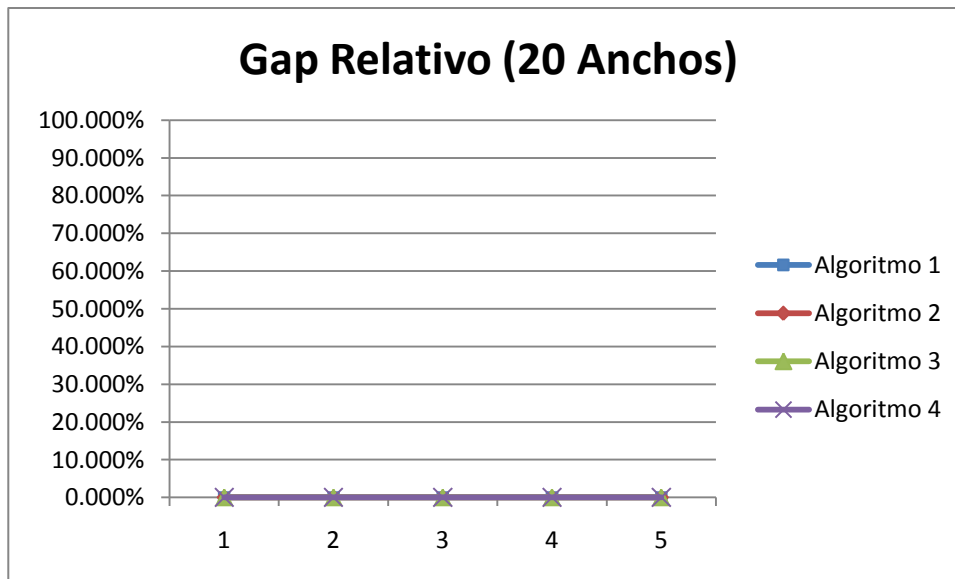


Ilustración 6.-Comparación del Gap Relativo entre los 4 algoritmos (Elaboración Propia).

La tabla que se muestra a continuación resume los resultados obtenidos (tiempo de ejecución en segundos de CPU y número de iteraciones), para las instancias con 30 anchos diferentes (ver Anexo 2).

Tabla 5.- Tiempo de Ejecución y Número de Iteraciones para instancias con 30 anchos (Elaboración Propia).

	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones
1	2.058	41	1.401	69	1.997	37	2.127	52
2	2.559	59	1.499	70	1.621	37	1.229	55
3	2.328	51	1.82	75	1.399	36	0.995	45
4	2.119	55	2.089	76	1.887	40	1.151	44
5	2.295	47	0.955	62	1.356	31	2.06	38
Totales	11.359	253	7.764	352	8.26	181	7.562	234

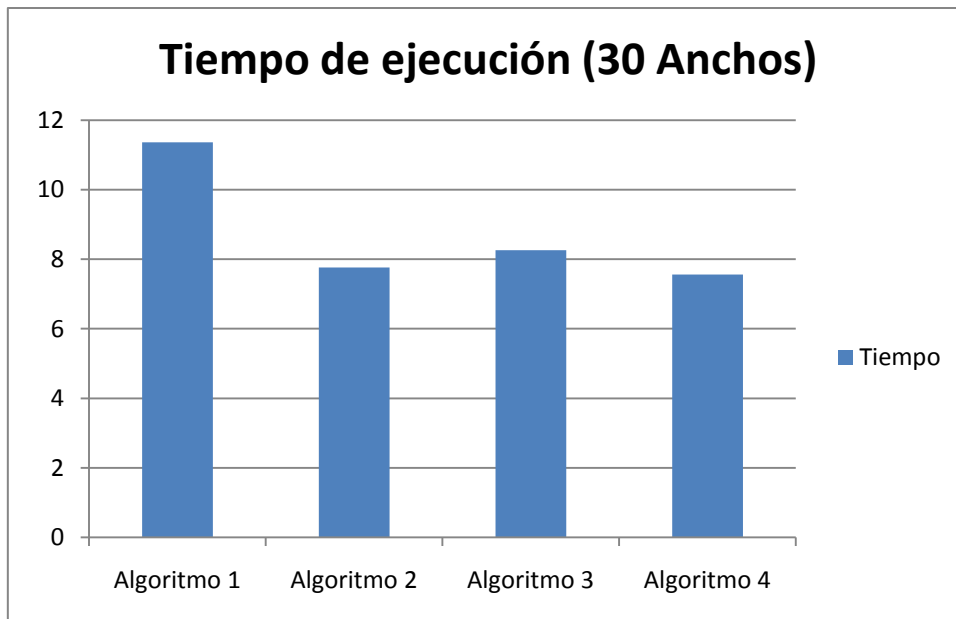


Ilustración 7.- Comparación entre los tiempos de ejecución para instancias con 30 anchos (Elaboración Propia).

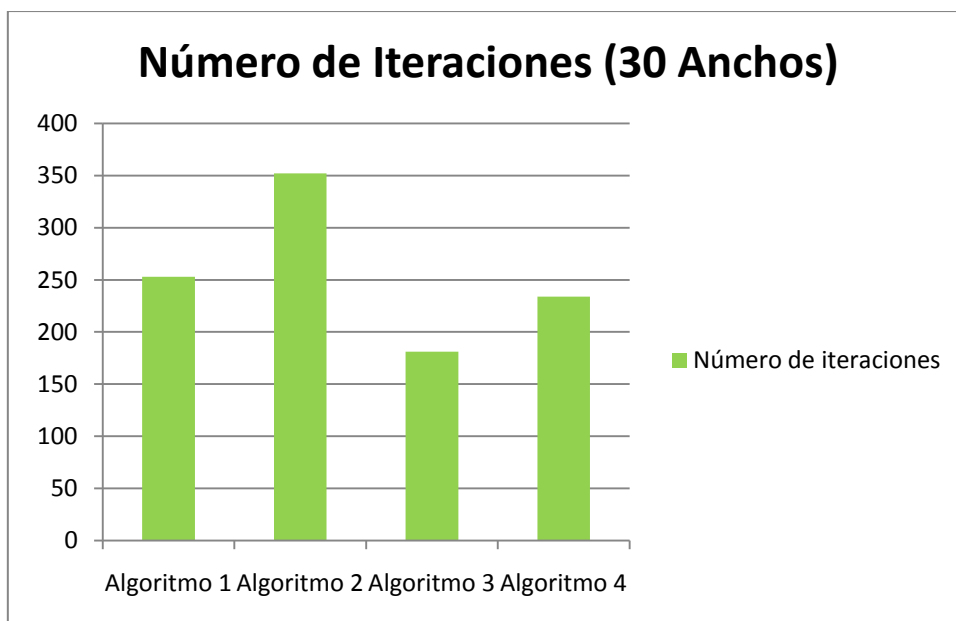


Ilustración 8.- Comparación entre el número de iteraciones para problemas con 30 anchos (Elaboración Propia).

Al comparar los tiempos de ejecución podemos observar una tendencia que a partir de ahora se irá notando más a medida que crece el número de anchos. Para el “Algoritmo 4” el tiempo de ejecución necesario para generar una solución inicial se compensa con el tiempo para generar patrones adicionales con la heurística del problema de la mochila y por tanto es el algoritmo con menor tiempo de ejecución. Dicho tiempo es 33% menor que el del “Algoritmo 1”.

Con respecto al número de iteraciones se repite el mismo patrón observado en las instancias del problema con 20 anchos, en donde el algoritmo con menor número de iteraciones es el “Algoritmo 3” (28% menos que el “Algoritmo 1”), siendo el

“Algoritmo 4” y el “Algoritmo 1” muy parecidos con respecto al número de iteraciones requeridas, y, finalmente, el algoritmo con mayor número de iteraciones, el “Algoritmo 2”.

En la Tabla 3 y la Ilustración 8 se puede observar el *gap* relativo para las instancias del problemas con 30. Se puede observar lo siguiente:

1. Hay una variación entre el *gap* relativo de los diferentes algoritmos evaluados, pero es tan pequeña, que la calidad de la solución no se deteriora con el uso de las heurísticas.
2. Podemos observar que el “Algoritmo 1” nos da una mejor calidad de solución con un promedio de *gap* relativo de 0.020%, mientras que el “Algoritmo 2” tiene un promedio de 0.029% y los “Algoritmos 3 y 4” tienen un promedio de 0.028%. Sin embargo es tan pequeña esta diferencia que no afecta de manera significativa la calidad de la solución.

Tabla 6.- Cota Inferior, Cota Superior y Gap Relativo para instancias de prueba con 30 anchos (Elaboración Propia).

	Cota Inferior	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
		Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	2493.77	2495	0.049%	2495	0.049%	2495	0.049%	2495	0.049%
2	2804.77	2805	0.000%	2805	0.000%	2806	0.044%	2806	0.044%
3	2798.22	2799	0.000%	2799	0.000%	2799	0.000%	2799	0.000%
4	2555.69	2557	0.051%	2557	0.051%	2556	0.000%	2556	0.000%
5	2747.77	2748	0.000%	2749	0.045%	2749	0.045%	2749	0.045%

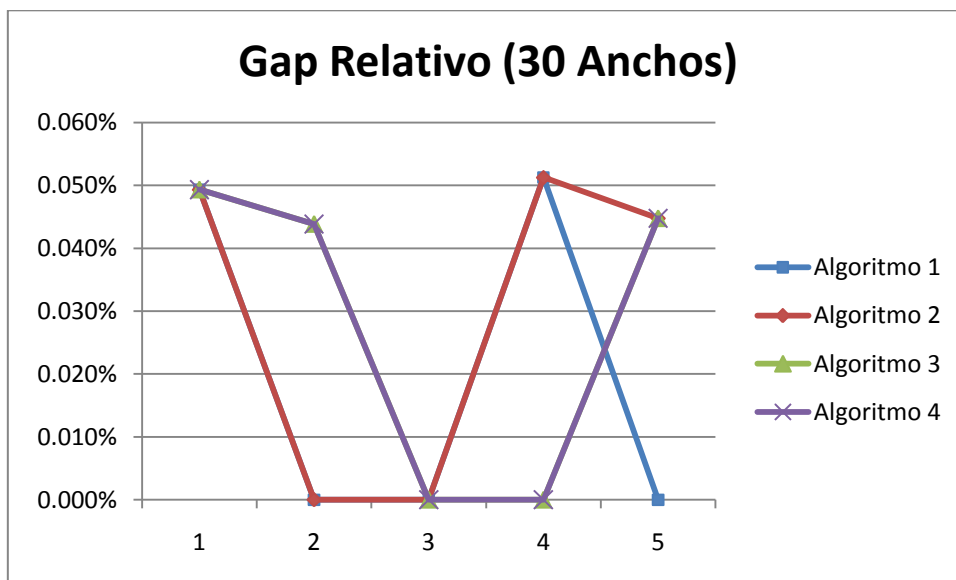


Ilustración 9.- Comparación del Gap Relativo entre los 4 algoritmos (Elaboración Propia).

La tabla que se muestra a continuación resume los resultados obtenidos (tiempo de ejecución en segundos de CPU y número de iteraciones), de las instancias del problema con 40 anchos (ver Anexo 2).

Tabla 7.- Tiempo de Ejecución y Número de Iteraciones para instancias con 40 anchos (Elaboración Propia).

	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones
1	4.181	89	4.274	150	3.245	68	2.605	111
2	4.696	67	2.293	118	4.726	49	2.48	82
3	4.368	104	2.933	140	3.182	70	2.574	103
4	1.154	36	0.296	46	0.188	6	0.094	8
5	2.714	74	2.372	122	2.106	56	1.108	70
Totales	17.113	370	12.168	576	13.447	249	8.861	374

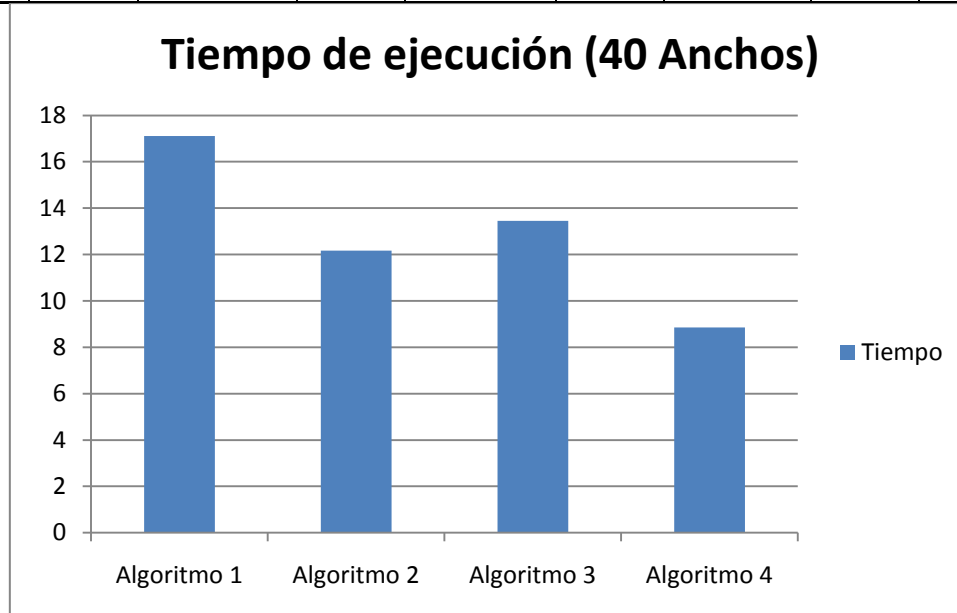


Ilustración 10.- Comparación entre los tiempos de ejecución para instancias con 40 anchos (Elaboración Propia).

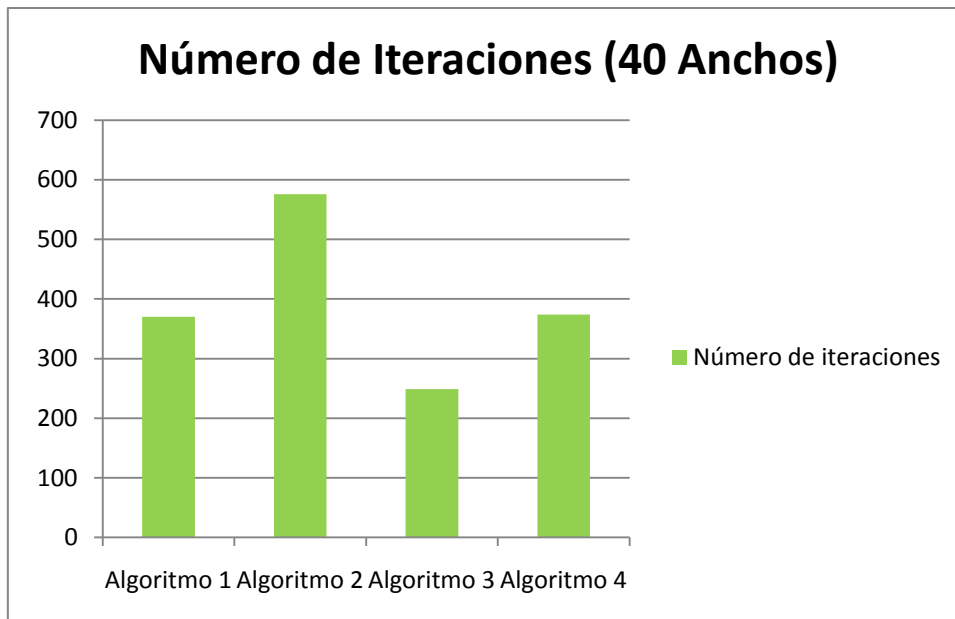


Ilustración 11.- Comparación entre el número de iteraciones para instancias con 40 anchos (Elaboración Propia).

Si observamos el tiempo de ejecución, la tendencia vista anteriormente se acentúa, siendo el “Algoritmo 4” el que requiere el menor tiempo de ejecución, siendo 48% más rápido que el “Algoritmo 1”. Al comparar el número de iteraciones, podemos observar que se repite el mismo patrón que se ha ido observando para las instancias del problema con 20 anchos o más. El “Algoritmo 3”, requiere el menor número de iteraciones (casi 33% iteraciones menos que el “Algoritmo 1”), y el “Algoritmo 2” es el que requiere el mayor número de iteraciones.

Al igual que en las instancias del problema con 30 anchos, cuando comparamos el *gap* relativo para instancias del problema con 40 anchos (ver

	Cota Inferior	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
		Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	3896.14	3897	0.000%	3897	0.000%	3897	0.000%	3897	0.000%
2	3684.99	3686	0.027%	3687	0.055%	3686	0.027%	3686	0.027%
3	4232.32	4233	0.000%	4233	0.000%	4233	0.000%	4233	0.000%
4	3711.00	3712	0.000%	3712	0.000%	3713	0.054%	3713	0.054%
5	4652.38	4653	0.000%	4653	0.000%	4653	0.000%	4653	0.000%

y Ilustración 12) hay una variación con respecto a las soluciones factibles obtenidas con los diferentes algoritmos de generación de columnas. Y de nueva cuenta el “Algoritmo 1” nos genera las soluciones factibles con mejor calidad siendo el promedio del *gap* relativo de 0.005%, pero de nueva cuenta podemos que la diferencia entre los *gap* relativos no es significativa, siendo el promedio del “Algoritmo 2” de 0.011% y el del los “Algoritmos 3 y 4” 0.016%.

Tabla 8.- Cota Inferior, Cota Superior y Gap Relativo para instancias de prueba con 40 anchos (Elaboración Propia).

Cota	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
------	-------------	-------------	-------------	-------------

	Inferior	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	3896.14	3897	0.000%	3897	0.000%	3897	0.000%	3897	0.000%
2	3684.99	3686	0.027%	3687	0.055%	3686	0.027%	3686	0.027%
3	4232.32	4233	0.000%	4233	0.000%	4233	0.000%	4233	0.000%
4	3711.00	3712	0.000%	3712	0.000%	3713	0.054%	3713	0.054%
5	4652.38	4653	0.000%	4653	0.000%	4653	0.000%	4653	0.000%

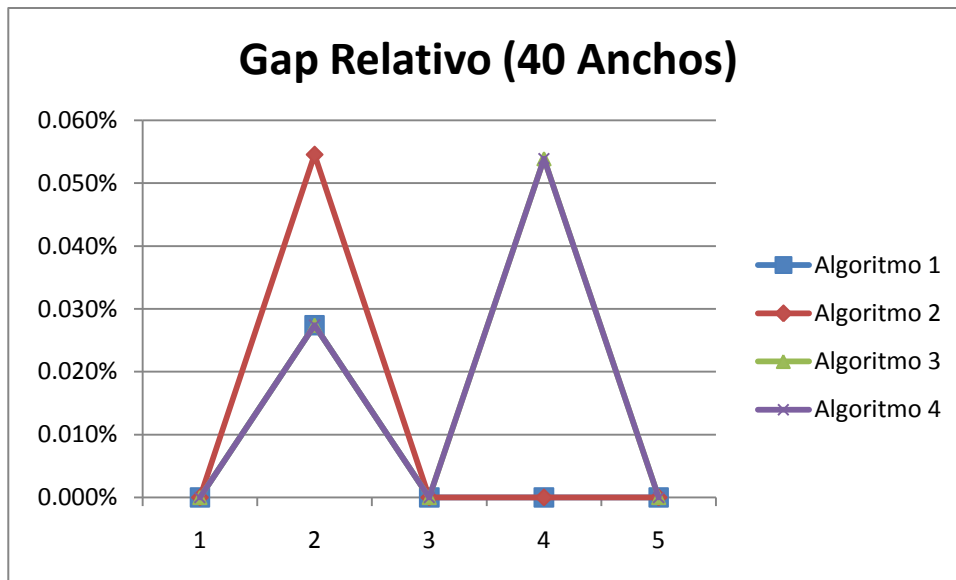


Ilustración 12.- Comparación del Gap Relativo entre los 4 Programas (Elaboración Propia).

La Tabla 11 muestra los resultados obtenidos (tiempo de ejecución en segundos de CPU y número de iteraciones), de las instancias del problema con 50 anchos (ver Anexo 2).

Tabla 9.- Tiempo de Ejecución y Número de Iteraciones para instancias con 50 anchos (Elaboración Propia).

	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones	Tiempo	Iteraciones
1	9.064	85	3.354	149	4.586	54	1.201	83
2	8.751	101	4.244	150	17.27	72	4.04	102
3	3.479	80	1.606	137	3.681	58	2.824	75
4	5.054	79	1.935	114	2.481	45	1.622	65
5	19.781	65	12.292	115	1.451	40	0.952	48
Totales	46.129	410	23.431	665	29.469	269	10.639	373

		r	o	r	o	r	o	r	o
1	5213.86	5215	0.022%	5215	0.022%	5216	0.041%	5216	0.041%
2	4879.09	4880	0.000%	4880	0.000%	4880	0.000%	4881	0.039%
3	4525.20	4527	0.040%	4527	0.040%	4527	0.040%	4527	0.040%
4	4865.31	4867	0.035%	4868	0.055%	4868	0.055%	4868	0.055%
5	3911.52	3914	0.063%	3915	0.089%	3917	0.140%	3915	0.089%

y Ilustración 15) se vuelve a observar que hay una variación con respecto al *gap* relativo. Podemos observar que mientras va creciendo el problema se acentúa la diferencia en la calidad de la solución, siendo el promedio del “Algoritmo 1” de 0.031%, del “Algoritmo 2” de 0.041%, del “Algoritmo 3” de 0.055% y del “Algoritmo 4” de 0.053%, esto nos sugiere que mientras va creciendo el número de anchos se sacrifica la calidad de la solución. Sin embargo, esta variación es tan pequeña que no es significativa.

Tabla 10.- Cota Inferior, Cota Superior y Gap Relativo para instancias de prueba con 40 anchos (Elaboración Propia).

	Cota Inferior	Algoritmo 1		Algoritmo 2		Algoritmo 3		Algoritmo 4	
		Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo	Cota Superior	Gap Relativo
1	5213.86	5215	0.022%	5215	0.022%	5216	0.041%	5216	0.041%
2	4879.09	4880	0.000%	4880	0.000%	4880	0.000%	4881	0.039%
3	4525.20	4527	0.040%	4527	0.040%	4527	0.040%	4527	0.040%
4	4865.31	4867	0.035%	4868	0.055%	4868	0.055%	4868	0.055%
5	3911.52	3914	0.063%	3915	0.089%	3917	0.140%	3915	0.089%

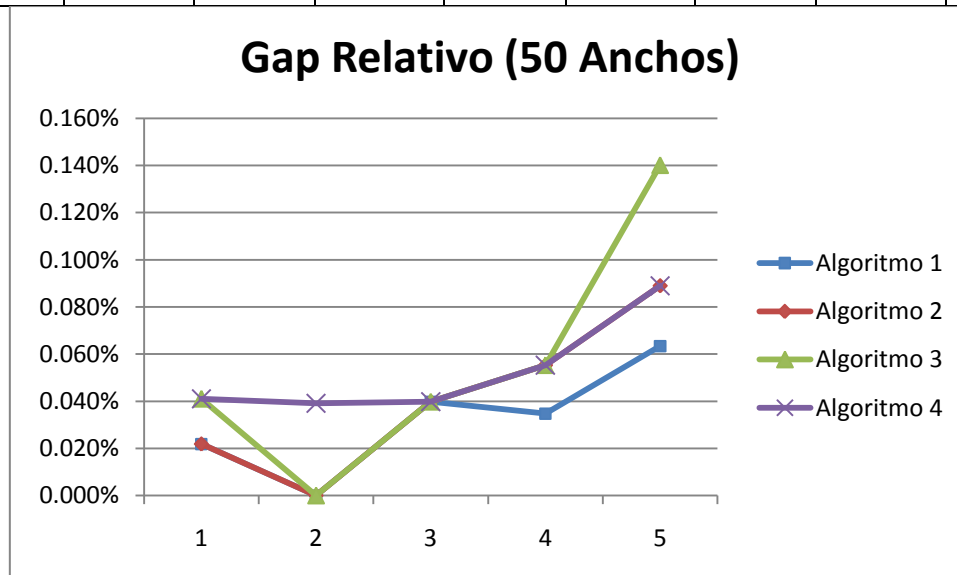


Ilustración 15.- Comparación del Gap Relativo entre los 4 algoritmos (Elaboración Propia).

4.2 Tendencias de Comportamiento de los Programas

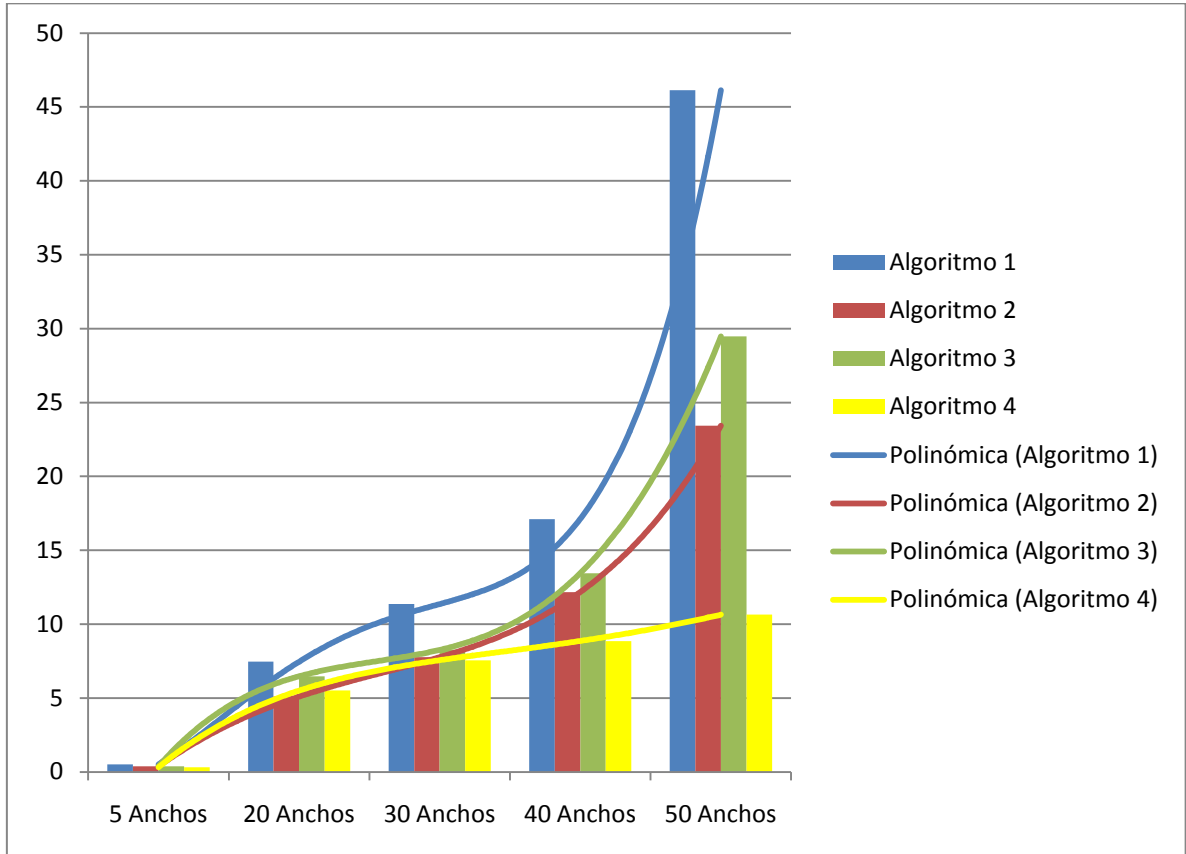


Ilustración 16.- Tendencias de crecimiento del tiempo de ejecución (Elaboración Propia).

Como podemos observar en la Ilustración 16, hay una diferencia significativa en la tendencia del tiempo de ejecución para los distintos algoritmos de generación de columnas para el problema de corte. Mientras que en los algoritmos 1, 2 y 3 podemos observar un crecimiento uniforme y exponencial, a partir de problemas con más de 20 anchos, el “Algoritmo 4” parece tener un crecimiento en tiempo de ejecución, en función del número de anchos, más amortiguado. Esto es una mejora significativa en comparación con los otros 3 programas, debido a que aunque el tiempo de ejecución crece a medida que aumenta el número de anchos, dicho crecimiento es menor que para el resto de los algoritmos

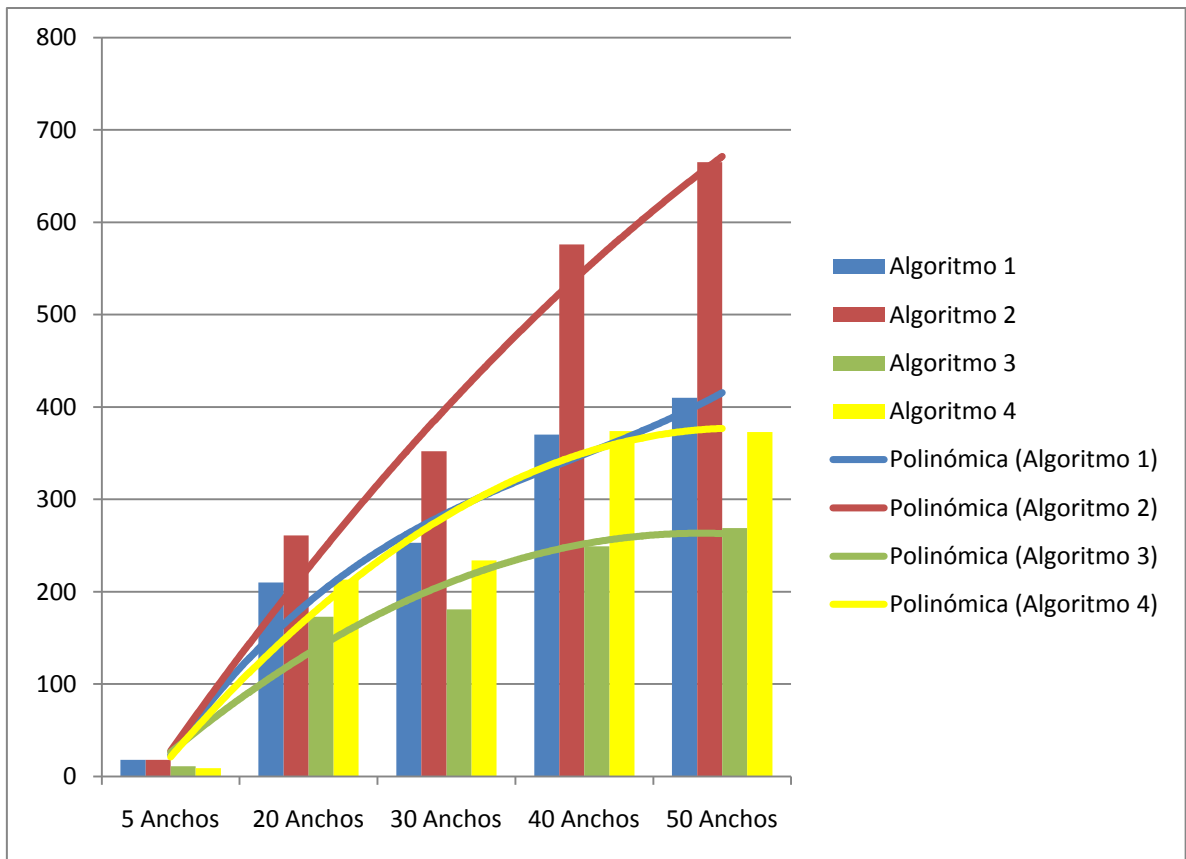


Ilustración 17.- Tendencias de crecimiento del número de iteraciones (Elaboración Propia).

Podemos observar en la Ilustración 17, que todos los algoritmos tienen una tendencia de crecimiento uniforme en el número de iteraciones a medida que aumenta el número de anchos, parecido a la función de la raíz cuadrada. Esto sugiere que mientras el problema sea más complejo el número de iteraciones va a crecer paulatinamente, y no se observará una diferencia significativa entre el crecimiento del número de iteraciones de los diferentes algoritmos.