PROYECTO ALGORITMOS DE ORDENAMIENTO.

Sorting Algorithms Proyect

RESUMEN
En este trabajo se van a aplicar diferentes algoritmos de ordenamiento y a comparar sus resultados y tiempos de ejecución para diferentes valores.

PALABRAS CLAVES: Algoritmo, Ordenamiento, Implementar, Método.

ABSTRACT
In this paper we will apply different sorting algorithms and compare their results and execution times for different values.

KEYWORDS: *Algorithm, Implement, Method, Sorting.*

DOUGLAS HERNANDEZ
Ingeniería de Sistemas y Computación
Estudiante
Universidad Tecnológica de Pereira
fullarukad@hotmail.com

HUGO ANDRES GOMEZ
Ingeniería de Sistemas y Computación
Estudiante
Universidad Tecnológica de Pereira
sureno755@gmail.com

INTRODUCCIÓN

Uno de los principales problemas en la ciencia de la computación está relacionado con el ordenamiento de un conjunto de elementos. Para esto existen bastantes métodos, algunos sencillos, como el de la burbuja, y otros más complicados, como el de conteo, con la ventaja de entregar resultados en menor tiempo.

Éstos a su vez, de acuerdo a su complejidad, pueden ser divididos en diferentes clases, denotándose con la notación Big-O.

En este trabajo se presentan algunos de los algoritmos de ordenamiento más comunes, entre los cuales tenemos:

- BubbleSort e InsertionSort: complejidad $O(n^2)$ que significa que el algoritmo tiene una complejidad potencial.
- CountingSort: complejidad O(n), que significa que el algoritmo tiene una complejidad lineal.
- QuickSort: complejidad O(nlogn), que significa que el algoritmo tiene una complejidad logarítmica.

ANÁLISIS DE LOS ALGORITMOS

BubbleSort: Es el método más sencillo y antiguo para ordenar un conjunto de datos, a su vez es el más lento.

Entrada (n miles)	Tiempo Real (ms)	Complejidad ($\Theta(n^2)$)	Factor Constante
20	1654	400	4,135
20	1666	400	4,165
40	6910	1600	4,31875
40	6905	1600	4,315625
60	15693	3600	4,359166667
60	15698	3600	4,360555556
80	27830	6400	4,3484375
80	27846	6400	4,3509375
100	43758	10000	4,3758
100	43766	10000	4,3766
120	63071	14400	4,379930556
120	63050	14400	4,378472222
140	86128	19600	4,394285714
140	86136	19600	4,394693878
160	113521	25600	4,434414063
160	113535	25600	4,434960938
180	147326	32400	4,547098765
180	147340	32400	4,547530864
200	185094	40000	4,62735
200	185090	40000	4,62725
220	227432	48400	4,699008264
220	227450	48400	4,699380165
240	272920	57600	4,738194444
240	272907	57600	4,73796875
260	324449	67600	4,79954142
260	324430	67600	4,799260355
280	375102	78400	4,784464286
280	375130	78400	4,784821429
300	433815	90000	4,820166667
300	433840	90000	4,820444444

Tabla 1. BubbleSort aplicado con diferentes tamaños de datos.

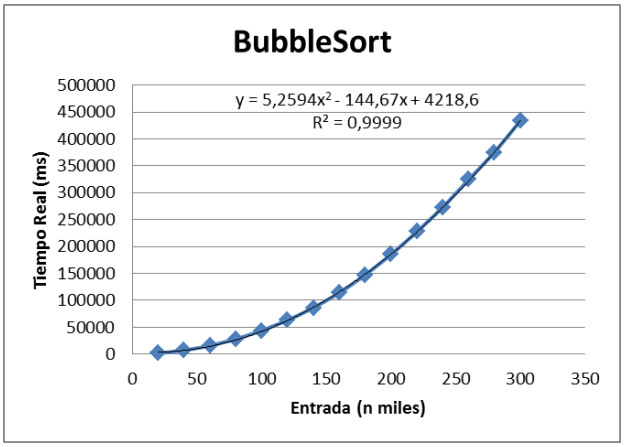

Figura 1. Gráfica del BubbleSort

InsertionSort: El Insertion Sort trabaja insertando el ítem en su lugar correspondiente al final de la lista.

Entrada (n miles)	Tiempo Real (ms)	Complejidad (Θ(n²))	Factor Constante
20	172	400	0,43
20	172	400	0,43
40	765	1600	0,478125
40	777	1600	0,485625
60	1934	3600	0,53722222
60	1935	3600	0,5375
80	3620	6400	0,565625
80	3622	6400	0,5659375
100	5803	10000	0,5803
100	5803	10000	0,5803
120	8534	14400	0,59263889
120	8533	14400	0,59256944
140	11700	19600	0,59693878
140	11716	19600	0,5977551
160	15552	25600	0,6075
160	15516	25600	0,60609375
180	19905	32400	0,61435185
180	19920	32400	0,61481481
200	25116	40000	0,6279
200	25121	40000	0,628025
220	31278	48400	0,64623967
220	31268	48400	0,64603306
240	38920	57600	0,67569444
240	38913	57600	0,67557292
260	45474	67600	0,67269231
260	45480	67600	0,67278107
280	53883	78400	0,68728316
280	53876	78400	0,68719388
300	62821	90000	0,69801111
300	62829	90000	0,6981

Tabla 2. InsertionSort aplicado con diferentes tamaños de datos.

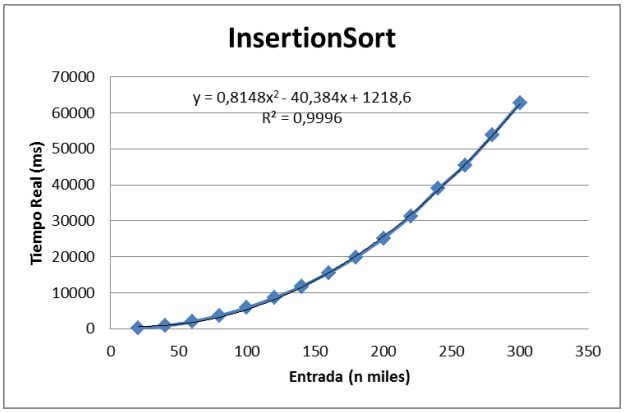

Figura 2. Gráfica del InsertionSort

QuickSort: es un algoritmo del estilo divide y vencerás.

Entrada (n miles)	Tiempo Real (ms)	Complejidad (Θ(nlogn))	Factor Constante
100	13	460,5170186	0,0282291
100	14	460,5170186	0,0304006
200	23	1059,663473	0,021705
200	24	1059,663473	0,0226487
300	35	1711,134742	0,0204543
300	36	1711,134742	0,0210387
400	46	2396,585819	0,019194
400	47	2396,585819	0,0196112
500	58	3107,304049	0,0186657
500	58	3107,304049	0,0186657
600	69	3838,157793	0,0179774
600	70	3838,157793	0,0182379
700	80	4585,756235	0,0174453
700	81	4585,756235	0,0176634
800	92	5347,689382	0,0172037
800	94	5347,689382	0,0175777
900	104	6122,155287	0,0169875
900	105	6122,155287	0,0171508
1000	115	6907,755279	0,016648
1000	117	6907,755279	0,0169375
1100	127	7703,372005	0,0164863
1100	129	7703,372005	0,0167459
1200	139	8508,092203	0,0163374
1200	141	8508,092203	0,0165725
1300	152	9321,155406	0,016307
1300	153	9321,155406	0,0164143
1400	164	10141,91852	0,0161705
1400	167	10141,91852	0,0164663
1500	176	10969,83058	0,016044
1500	177	10969,83058	0,0161352

Tabla 3. QuickSort aplicado con diferentes tamaños de datos.

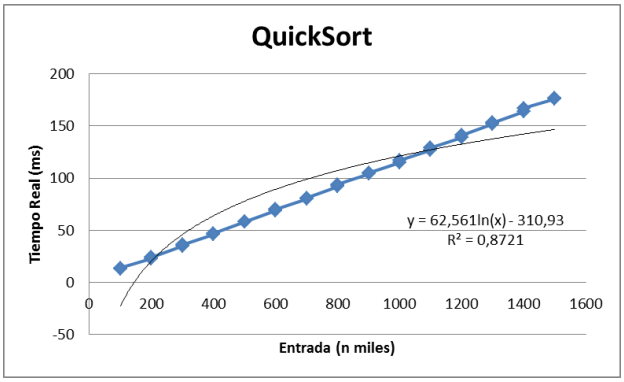

Figura 3. Gráfica del QuickSort.

CountingSort: se cuenta el número de elementos de cada clase para luego ordenarlos. Sólo puede ser utilizado para ordenar elementos que sean contables.

Entrada (n miles)	Tiempo Real (ms)	Complejidad (Θ(n+k))	Factor Constante
1000	58	6001000	9,6651E-06
1000	59	6001000	9,8317E-06
2000	114	6002000	1,8994E-05
2000	117	6002000	1,9494E-05
3000	149	6003000	2,4821E-05
3000	152	6003000	2,5321E-05
4000	191	6004000	3,1812E-05
4000	194	6004000	3,2312E-05
5000	234	6005000	3,8968E-05
5000	238	6005000	3,9634E-05
6000	275	6006000	4,5788E-05
6000	278	6006000	4,6287E-05
7000	326	6007000	5,427E-05
7000	330	6007000	5,4936E-05
8000	384	6008000	6,3915E-05
8000	388	6008000	6,4581E-05
9000	422	6009000	7,0228E-05
9000	429	6009000	7,1393E-05
10000	473	6010000	7,8702E-05
10000	483	6010000	8,0366E-05
11000	511	6011000	8,5011E-05
11000	522	6011000	8,6841E-05
12000	560	6012000	9,3147E-05
12000	563	6012000	9,3646E-05
13000	600	6013000	9,9784E-05
13000	604	6013000	0,00010045
14000	653	6014000	0,00010858
14000	655	6014000	0,00010891
15000	678	6015000	0,00011272
15000	685	6015000	0,00011388

Tabla 4. CoutingSort aplicado con diferentes tamaños de datos.

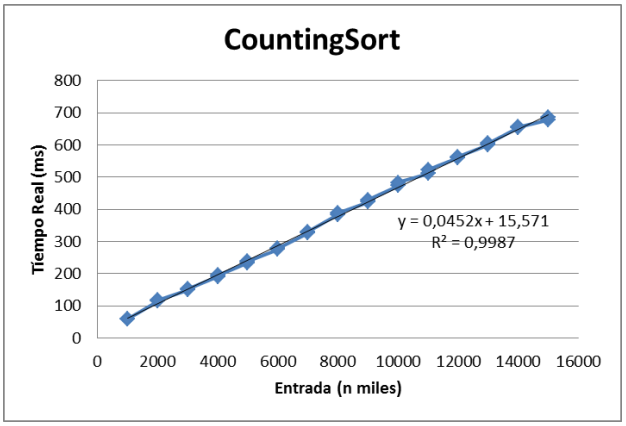

Figura 4. Gráfica del CoutingSort.

INFERIR TIEMPOS DE EJECUCION

Para hallar el tiempo estimado se utiliza la siguiente formula:

$$T = A*M$$

Donde "A" es el promedio del factor constante y "M" es la complejidad del algoritmo.

BubbleSort:

Entrada (n miles)	Tiempo Estimado (ms)	Tiempo Real (ms)
30	4067	3822
50	11296	10748
70	22141	21622
90	36600	35459
110	54674	53399
130	76363	74506
150	101666	99279
170	130585	130463
190	163118	166109
210	199266	206514
230	239029	249882
250	282406	297211
270	329399	348707
290	380006	406521
310	434228	461091

Tabla 5. Comparación Tiempo Estimado y Real.

InsertionSort:

Entrada (n miles)	Tiempo Estimado (ms)	Tiempo Real (ms)
30	466	390
50	1029	1217
70	2237	3120
90	4087	4664
110	6582	7504
130	9720	10499
150	13501	13806
170	17927	17877
190	22996	22651
210	28709	28361
230	35065	34695
250	42065	41886
270	49709	49780
290	57997	58329
310	66928	67657

Tabla 6. Comparación Tiempo Estimado y Real.

QuickSort:

Entrada (n miles)	Tiempo Estimado (ms)	Tiempo Real (ms)
150	13,9827144	18
250	25,6803799	28
350	38,1434428	39
450	51,1455288	50
550	64,5645098	66
650	78,3236349	75
750	92,370116	86
850	106,665391	97
950	121,180046	109
1050	135,890895	126
1150	150,779195	136
1250	165,829473	146
1350	181,028744	159
1450	196,365948	173
1550	211,831558	187

Tabla 7. Comparación Tiempo Estimado y Real.

CountingSort:

Entrada (n miles)	Tiempo Estimado (ms)	Tiempo Real (ms)
1500	0,09421429	87
2500	0,15702381	133
3500	0,21983334	177
4500	0,28264287	218
5500	0,34545239	266
6500	0,40826192	310
7500	0,47107144	371
8500	0,53388097	396
9500	0,5966905	438
10500	0,65950002	482
11500	0,72230955	524
12500	0,78511907	589
13500	0,8479286	629
14500	0,91073813	659
15500	0,97354765	700

Tabla 8. Comparación Tiempo Estimado y Real.

Usando la misma fórmula y sin necesidad de correr los algoritmos en el PC, evitando un posible bloqueo o que nos quedemos esperando una cantidad de tiempo inaceptable, se realizó para una cantidad de datos mucho mayor:

BubbleSort:

Entrada (n miles)	Tiempo Estimado (ms)
30000	4729124119
40000	8409257419
60000	18925164019
80000	33648590619
100000	52579537219
120000	75718003819
140000	103063990419
160000	134617497019
180000	170378523619
200000	210347070219

Tabla 9. Tiempo aproximado para cantidades muy grandes.

InsertionSort:

Entrada (n miles)	Tiempo Estimado (ms)
30000	732109699
40000	1302065859
60000	2930858179
80000	5211490499
100000	8143962819
120000	11728275139
140000	15964427459
160000	20852419779
180000	26392252099
200000	32583924419

Tabla 10. Tiempo aproximado para cantidades muy grandes.

QuickSort:

Entrada (n miles)	Tiempo Estimado (ms)
30000	334
40000	352
60000	377
80000	395
100000	409
120000	421
140000	430
160000	439
180000	446
200000	453

Tabla 11. Tiempo aproximado para cantidades muy grandes.

CountingSort:

Entrada (n miles)	Tiempo Estimado (ms)
30000	1372
40000	1824
60000	2728
80000	3632
100000	4536
120000	5440
140000	6344
160000	7248
180000	8152
200000	9056

Tabla 12. Tiempo aproximado para cantidades muy grandes.

CONCLUSIONES

Se puede notar que para algoritmos de complejidad O(n), de comportamiento lineal, su tiempo de ejecución es bastante rápido. Mientras que un algoritmo de complejidad O(nlogn) se puede demorar un poco más, así como también existen los de complejidad $O(n^2)$, los cuales suelen ser los más demorados.

Más allá de la complejidad de cada uno de los algoritmos, su eficiencia de ordenamiento se compara usando datos empíricos, pues su velocidad varía de acuerdo a las características del conjunto de datos a ordenar.

También hay que decir que no todos los algoritmos se comportan igual ante conjuntos de datos con características similares.