Práctica 1

Adam Bourbahh Romero

28/09/2025

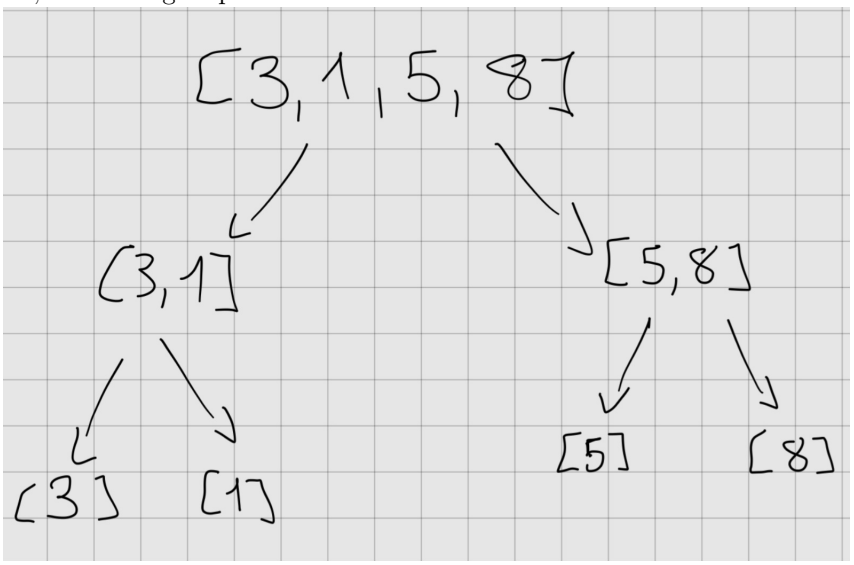
1 Análisis teórico del algoritmo Merge sort

Para este apartado he elegido la ordenación por mezcla. Este algoritmo recursivo consiste en la "división" recursiva de, por ejemplo, un array. Imaginemos que tenemos un array de 4 elementos **[3,1,5,8]** y queremos darle un orden ascendente.

1.1 División

Nuestro algoritmo lo que hace en este caso es dividir el array a la mitad de forma constante hasta tener cada elemento separado. Podemos intuir a partir de esto que el algoritmo va a hacer $\mathbf{log_2(n)}$ divisiones.

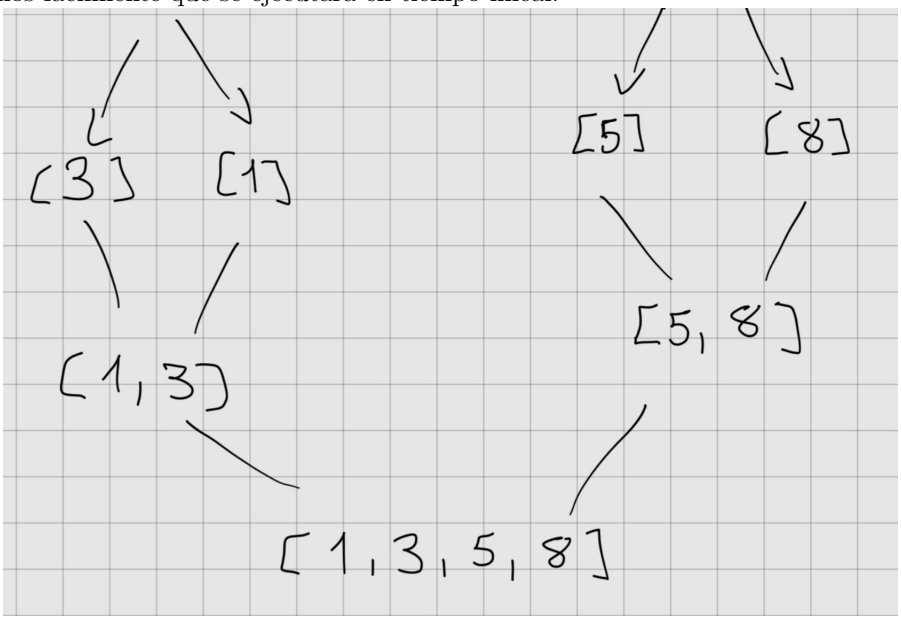
Con el ejemplo dado, nuestro algoritmo convierte **[3,1,5,8]** $\xrightarrow{1}$ **[3,1]** + **[5,8]** $\xrightarrow{2}$ **[3]**+**[1]** **[5]**+**[8]**. Podríamos pensar que en este proceso se hacen 3 operaciones, sin embargo aquí medimos el número de "niveles recursivos".



En el árbol que he adjuntado podemos ver como solo hay 2 niveles antes de llegar a las hojas del árbol.

1.2 Mezcla

La siguiente gran parte de este algoritmo consiste en ir de forma "ascendente" por el árbol para reordenar los subarrays, dado que en total hay n elementos, vemos fácilmente que se ejecutará en tiempo lineal.



1.3 Total

Al principio he adelantado que la complejidad del algoritmo es de **O(n*log n)**. Sin embargo podríamos pensar que por qué no es directamente O(n). Para clarificar esto, volvamos al esquema de árbol. Hemos dicho que teníamos log(n) niveles. Además, en cada uno de esos niveles se hacen n operaciones. Por ejemplo, en el último nivel de nuestro ejemplo se hacen una por cada rama, es decir, $1x4 = 4$ y en el segundo nivel, hacemos dos operaciones por cada rama, es decir, $2x2 = 4$. Por tanto, tenemos n operaciones en cada nivel :

$$\underbrace{\mathrm{n} + \mathrm{n} + \ldots + \mathrm{n}}_{log(n)\text{veces}}$$

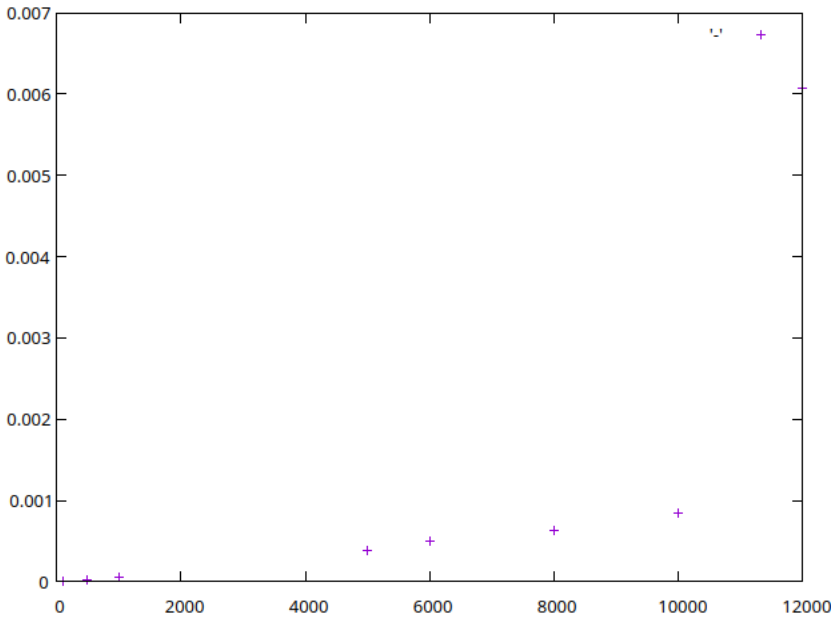
Equivalentemente, **nlog(n) operaciones.**

2 Cálculo de la eficiencia empírica

Una vez realizado el cálculo de teórico, vamos a ver que tanto se acerca a la eficiencia empírica. Habiendo ejecutado el código con distintos tamaños:

Nº Datos	T. Ejecución
100	4.918e-06
500	2.5652e-05
1000	5.3912e-05
5000	0.000389048
6000	0.000508194
8000	0.000627804
10000	0.000849083
12000	0.006077

A la hora de graficarlo podemos ver que el tiempo de ejecución se asemeja mucho a $\mathbf{f(x) = xlog(x)}$:



Sin embargo, la muestra de datos no es suficiente como para poder asegurarlo, por lo que usando el csh que se nos ha dado, lo he modificado para que funcione con el mergesort y he generado un archivo con cientos de ejecuciones. Tras ello, he usado la herramiento **fit** de gnuplot para averiguar el a y b que se adecue a $a * x * log(x) + b$. Por último, he hecho un plot de ambas gráficas y podemos ver un comportamiento claramente parecido. Adjunto a continuación las capturas:

```
gnuplot> f(x)=a*x*log(x)+b
gnuplot> fit f(x) "tiempos_busqueda.dat" via a,b
iter      chisq       delta/lim  lambda   a             b
   0 4.1764537494e+14   0.00e+00  4.57e+05    1.000000e+00   1.000000e+00
   1 1.0430728296e+08  -4.00e+11  4.57e+04    4.984421e-04   9.999987e-01
   2 2.7236695611e+02  -3.83e+10  4.57e+03   -1.306147e-06   9.999985e-01
   3 2.7235724084e+02  -3.57e+00  4.57e+02   -1.308628e-06   9.999855e-01
   4 2.7164787209e+02  -2.61e+02  4.57e+01   -1.306909e-06   9.986827e-01
   5 2.1255857295e+02  -2.78e+04  4.57e+00   -1.154803e-06   8.834363e-01
   6 1.0778347225e+00  -1.96e+07  4.57e-01   -7.206437e-08   6.307821e-02
   7 8.0058584394e-04  -1.35e+08  4.57e-02    1.085137e-08   2.554527e-04
   8 7.9995422268e-04  -7.90e+01  4.57e-03    1.091491e-08   2.073068e-04
   9 7.9995422268e-04  -4.65e-09  4.57e-04    1.091491e-08   2.073064e-04
iter      chisq       delta/lim  lambda   a             b

After 9 iterations the fit converged.
final sum of squares of residuals : 0.000799954
rel. change during last iteration : -4.64878e-14

degrees of freedom    (FIT_NDF)                      : 998
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 0.000895297
variance of residuals (reduced chisquare) = WSSR/ndf   : 8.01557e-07

Final set of parameters            Asymptotic Standard Error
=======================            ==========================
a               = 1.09149e-08      +/- 8.393e-11    (0.7689%)
b               = 0.000207306      +/- 5.424e-05    (26.16%)

correlation matrix of the fit parameters:
                a      b
a               1.000
b              -0.853  1.000
gnuplot> plot f(x)
         warning: Did you try to plot a complex-valued function?
QSocketNotifier: Can only be used with threads started with QThread
gnuplot> plot f(x), "tiempos_busqueda.dat"
```