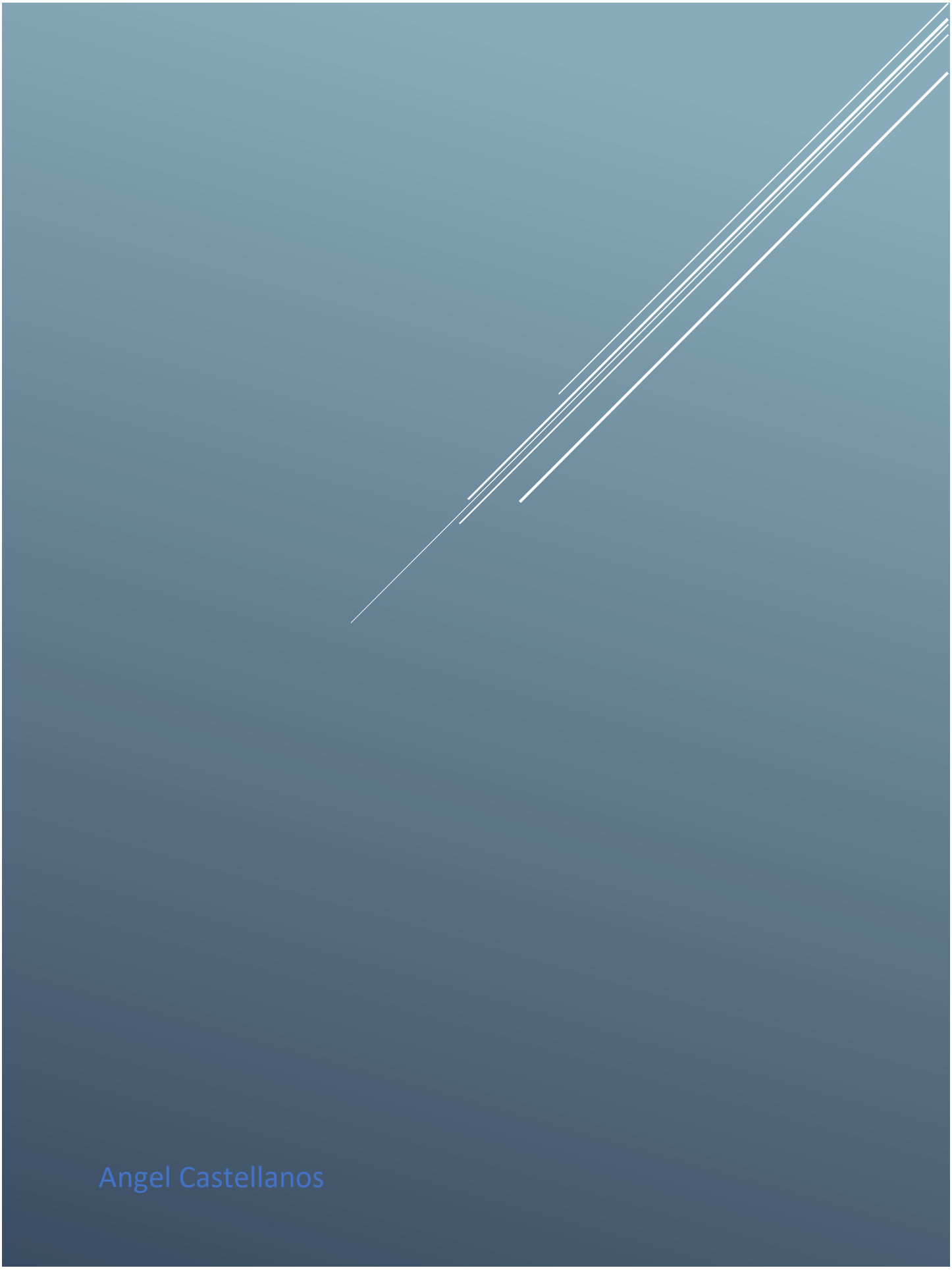
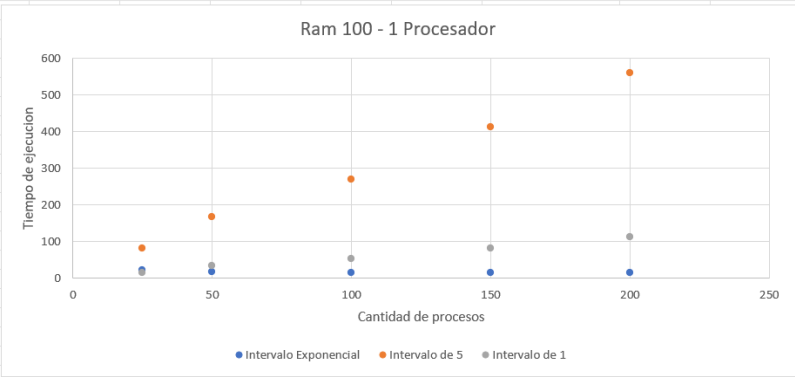
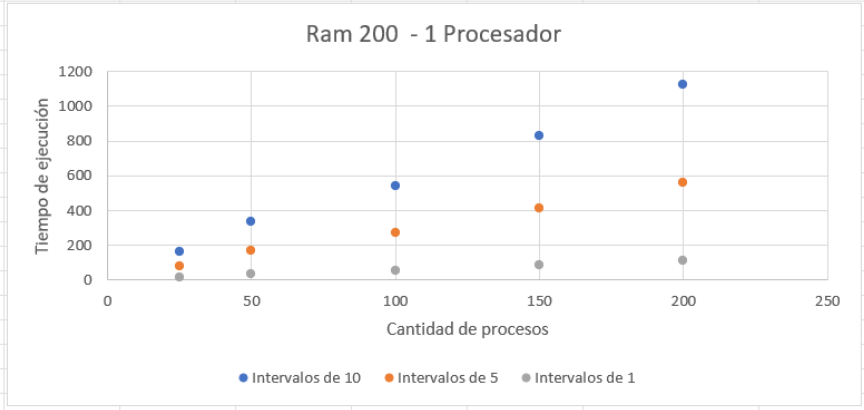

Angel Castellanos

Resultados:

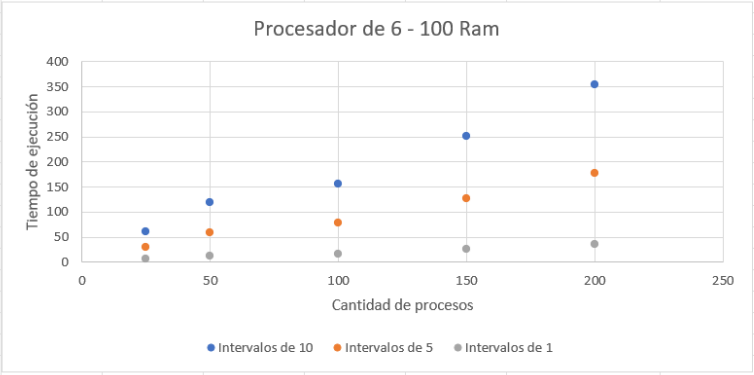
25	50	100	150	200	Desviación estándar
22.1980104	17.213952	14.1000446	14.6048187	14.0078407	3.482500199
81	167.4	269.65	413.666667	561.425	192.1708985
16.2	33.48	53.93	82.7333333	112.285	38.4341797



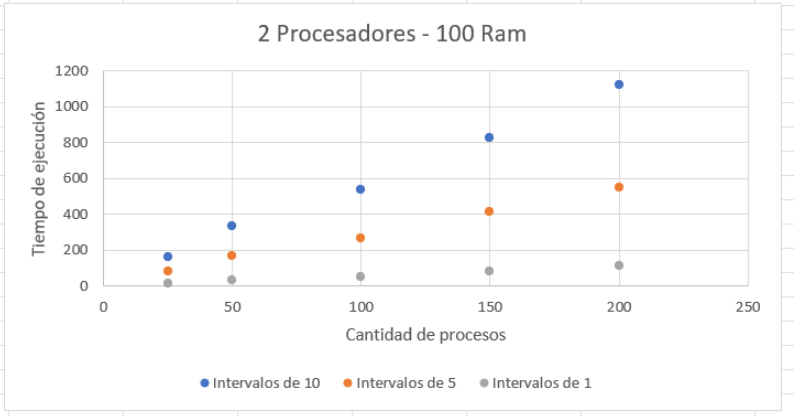
25	50	100	150	200	Desviación estándar
162	334.8	539.3	827.333333	1122.85	384.341797
81	167.4	269.65	413.666667	561.425	192.1708985
16.2	33.48	53.93	82.7333333	112.285	38.4341797



25	50	100	150	200	Desviación estándar
60.8	118.8	156.6	251.866667	354.8	116.0323097
30.4	59.4	78.3	125.933333	177.4	58.01615484
6.08	11.88	15.66	25.1866667	35.48	11.60323097



25	50	100	150	200	Desviación estándar
162	334.8	539.3	827.333333	1122.85	384.341797
81	167.4	269.65	413.666667	547.95	58.01615484
16.2	33.48	53.93	82.7333333	112.285	38.4341797



Luego de analizar los resultados esta claro que la manera más eficiente de tratar los datos es con un procesador rápido, que sea capaz de realizar varias instrucciones en un mismo ciclo, así como se ve en la gráfica 3, ya que presenta los menores tiempos para las diferentes cantidades de procesos. Así mismo también cabe resaltar que la llegada de los procesos en intervalos de 1 se posiciona como el mejor complemento para enviar los procesos, ya que reduce al mínimo el tiempo que el procesador pierde en esperar las instrucciones, es decir reduce el tiempo muerto, y por ende se saturan menos los datos.

Por esto la mejor estrategia para reducir el tiempo promedio, es utilizar un procesador rápido y un intervalo corto entre procesos.

Link Git-Hub

https://github.com/angelcast2002/HDT5