PART 5:

->After performing the respective tests, we have been able to show that parallelizing the first loop gives better results compared to the second, this is because if we parallelize the second lop, what we are doing is creating and destroying threads in each of the iterations of the first loop, which is not very optimal

->In our case, we have obtained similar performances in the different tests carried out, so we cannot position ourselves on which scheduling is better or worse.

* static: assignment determined at the beginning (doesn’t change while running)
* dynamic: assignment is carried out on-line
* guided: assignment is carried out on-line
* runtime: decision deferred to runtime (run-sched-var/OMP\_SCHEDULE)

#pragma omp parallel for schedule(static, chunk\_size)

#pragma omp parallel for schedule(dynamic, chunk\_size)

#pragma omp parallel for schedule(guided)

#pragma omp parallel for schedule(runtime)

|  |  |
| --- | --- |
| Scheduling (100 ITERATIONS) | Time (us) |
| Static 1 | 694059 |
| Static 5 | 702670 |
| Static 10 | 691784 |
| Dynamic 1 | 716381 |
| Dynamic 5 | 692066 |
| Dynamic 10 | 684185 |
| Guided | 699284 |
| Runtime | 691414 |

|  |  |
| --- | --- |
| Scheduling (1000 ITERATIONS) | Time (us) |
| Static 1 | 7051178 |
| Static 5 | 6946560 |
| Static 10 | 6942459 |
| Dynamic 1 | 7071090 |
| Dynamic 5 | 7065223 |
| Dynamic 10 | 6914113 |
| Guided | 7026037 |
| Runtime | 6807883 |

-> In our case, if we use private variables for width or height it does not work, because the same value will be used in each pixel of the image.