

Mini-Project (Part 1)

Parameters (default in code):

- pRE = 1000
- pIM = 1000
- iterations = 100
- threshold = 2
- Chunk Size Range = (1 to 200, step 10)

Performance Comparison

Computation specs:

- Intel® Core™ i5-11300H-processor

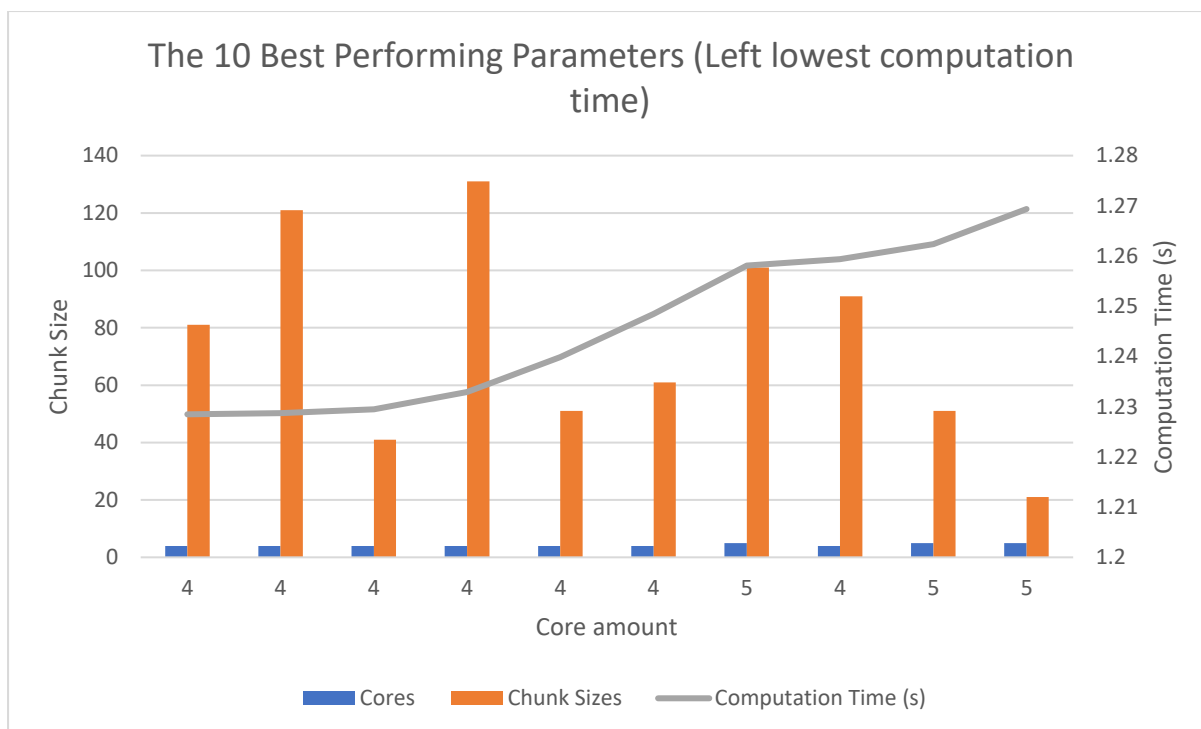
Algorithm	Script location (root)	Computation Time (s)
Naive	<i>mandelbrot_naive.py</i>	3.91s
Numpy vectorized	<i>mandelbrot_numpy.py</i>	2.12s
Numba-optimized	<i>mandelbrot_numba.py</i>	1.39s
Multiprocessing	<i>mandelbrot_multicore.py</i>	1.23s

Analysis of Multiprocessing Implementation

The 10 best performance results for different chunk sizes and processor amount

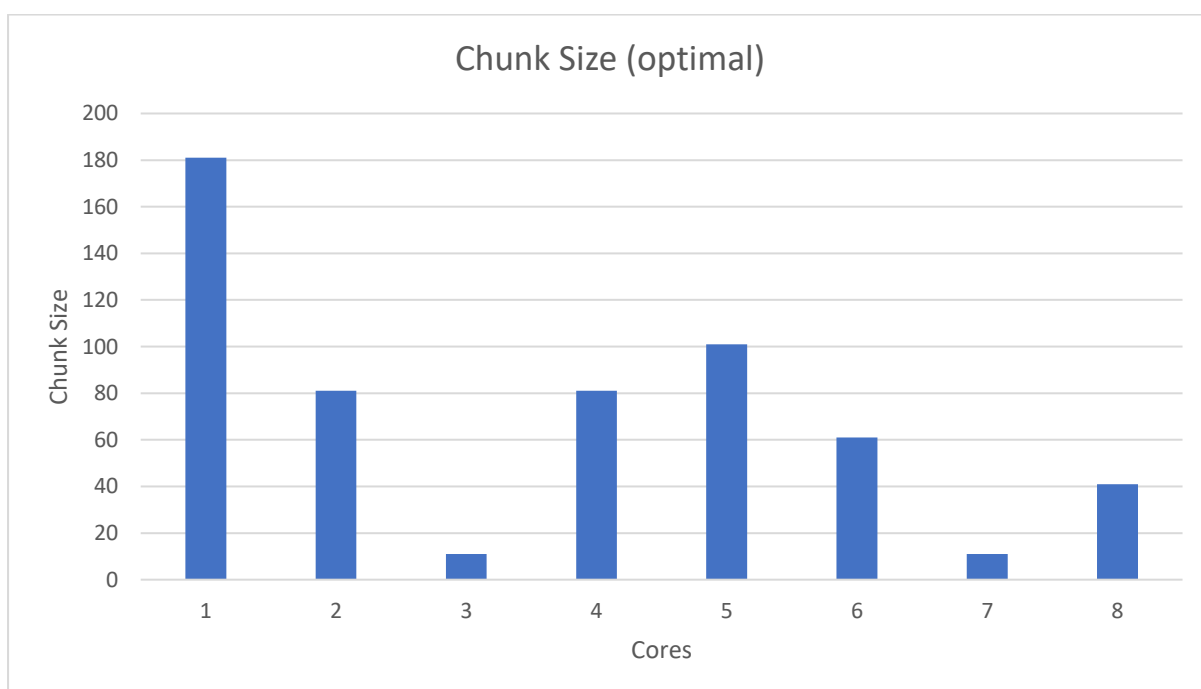
(See *Computations.xlsx*, Sheet: "Best Runs")

#Rank	Number of processors	Chunk Size	Computation Time (s)
1	4	81	1.22848
2	4	121	1.22871
3	4	41	1.2295
4	4	131	1.23295
5	4	51	1.23985
6	4	61	1.24845
7	5	101	1.2581
8	4	91	1.25935
9	5	51	1.26235
10	5	21	1.26936



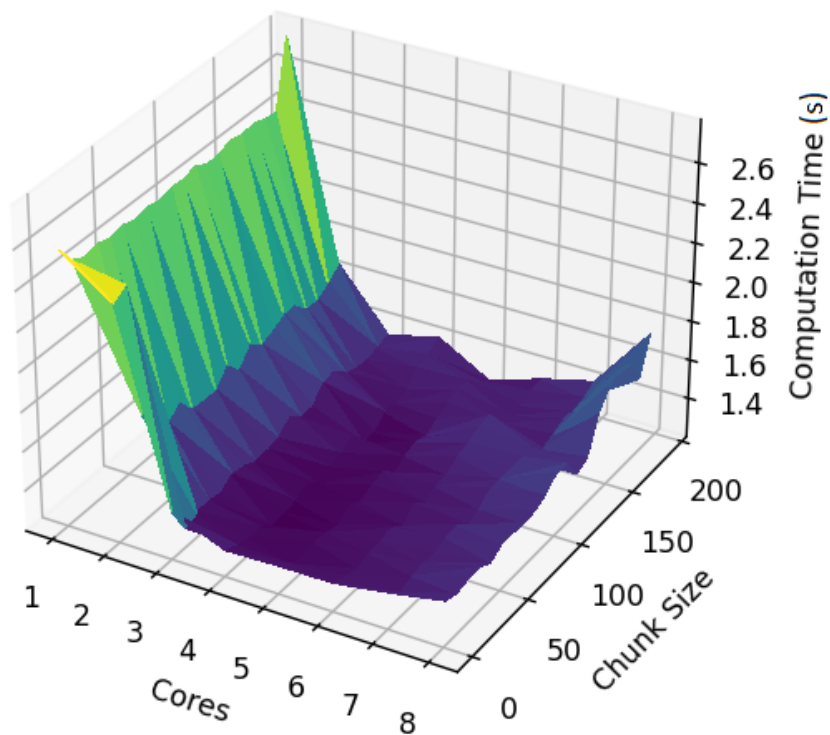
Optimal chunk size in relation to number of processes based on computation time:

(See Computations.xlsx, Sheet: "Optimal chunk size for core")

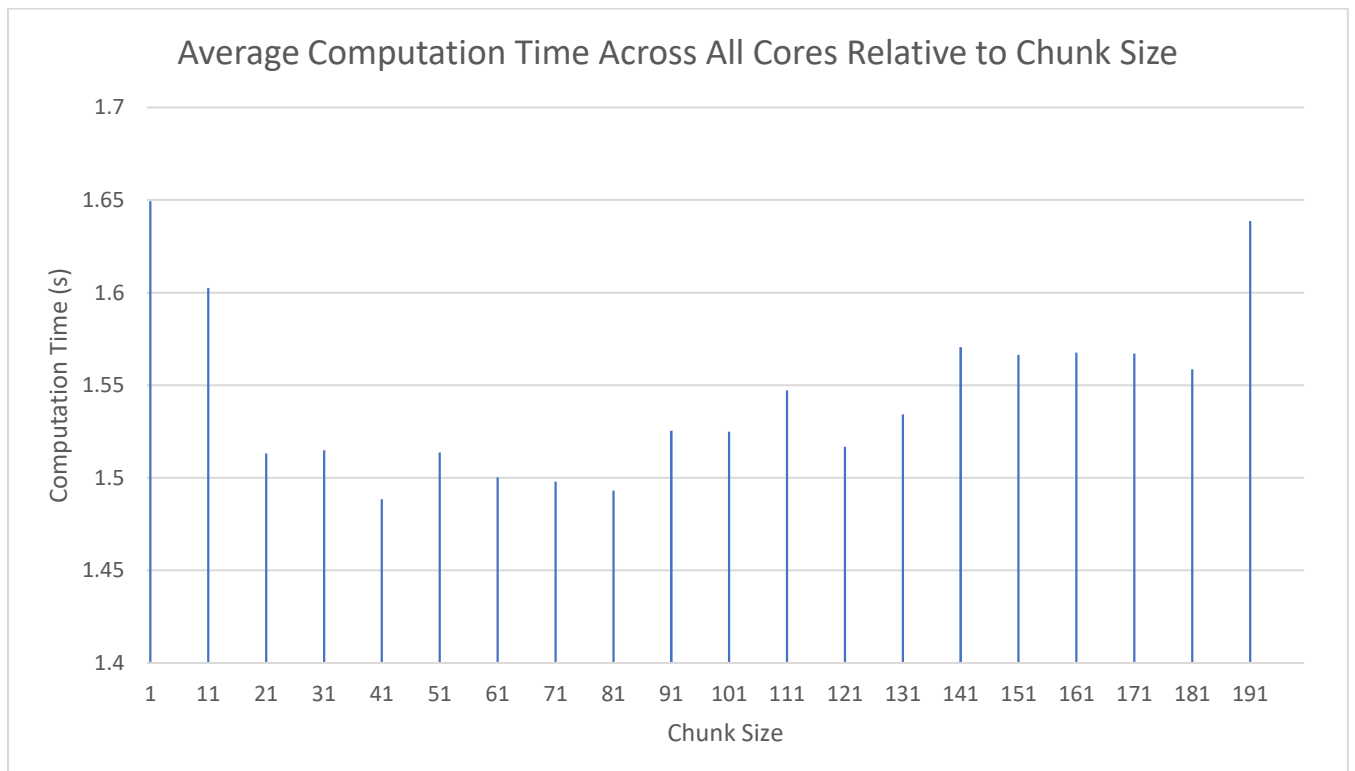


Gradient of computation time across parameters

(Graph is computed after running "mandelbrot_multicore.py")

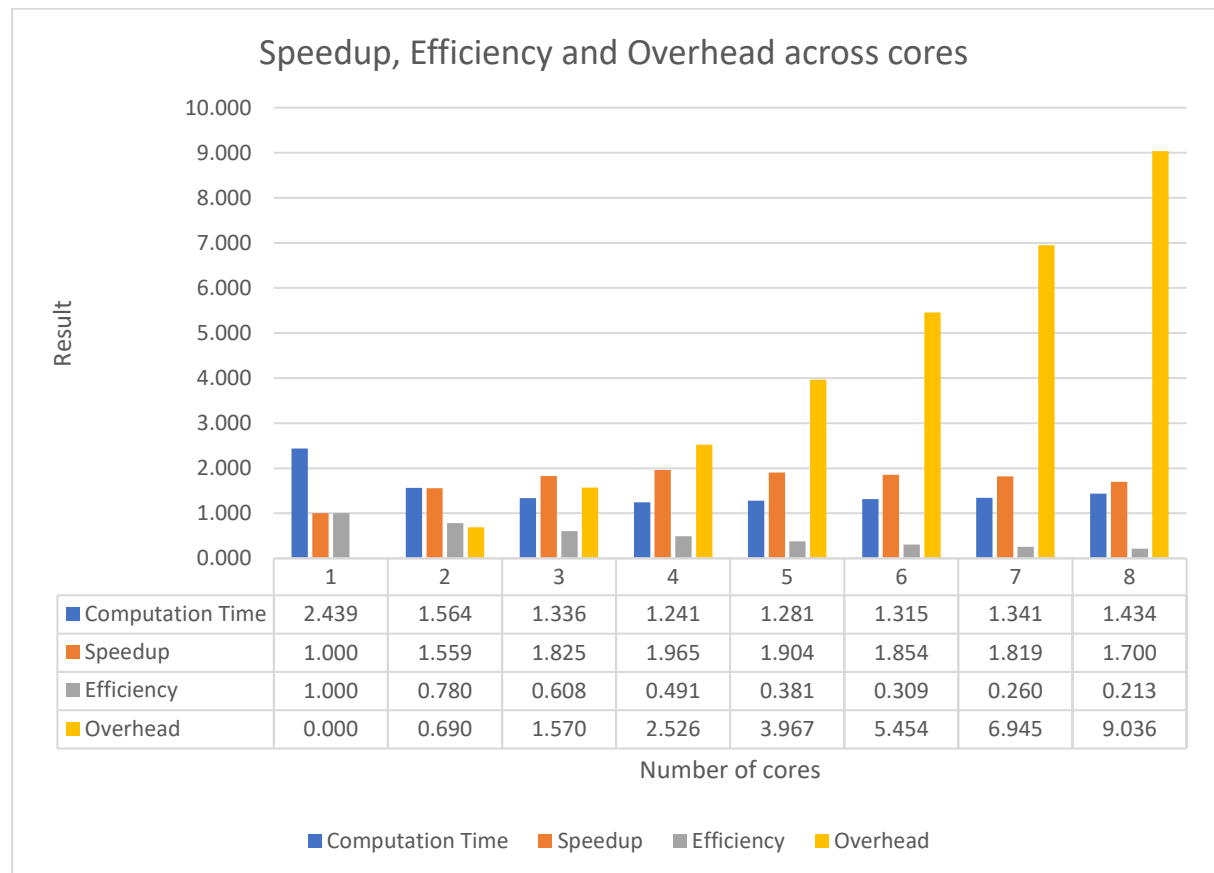


(See Computations.xlsx, Sheet: "Chunk Size")



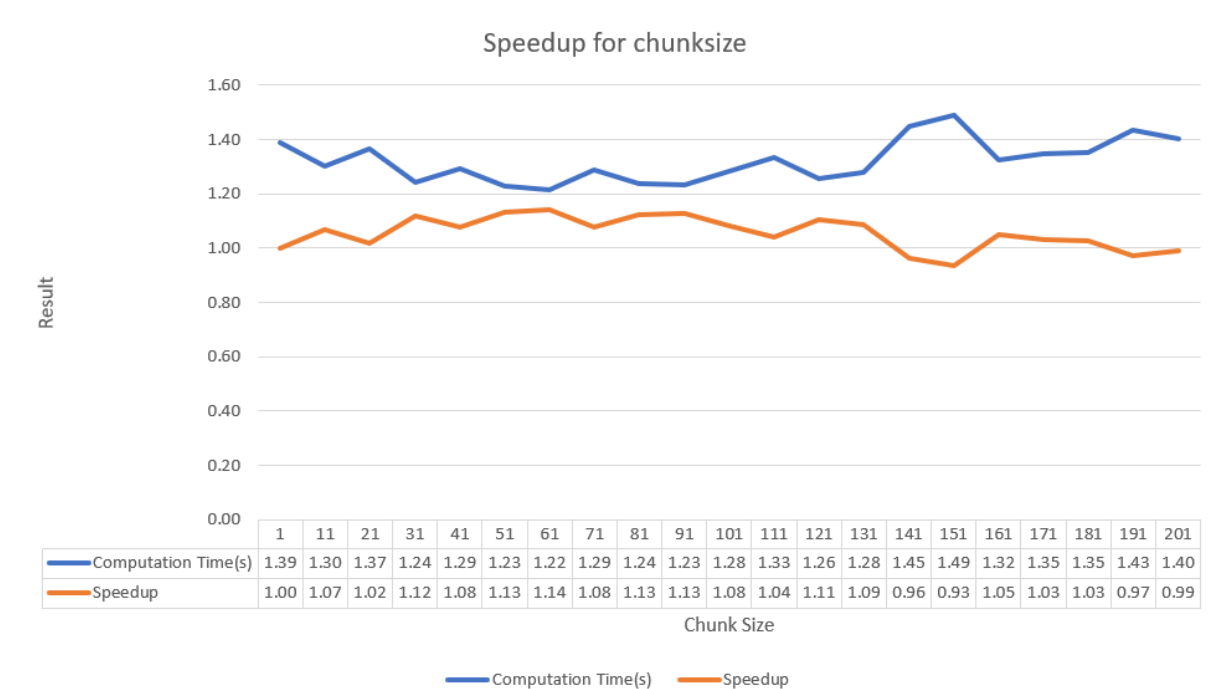
Speedup, Efficiency and Overhead across number of cores

(See Computations.xlsx, Sheet: "Speedup")

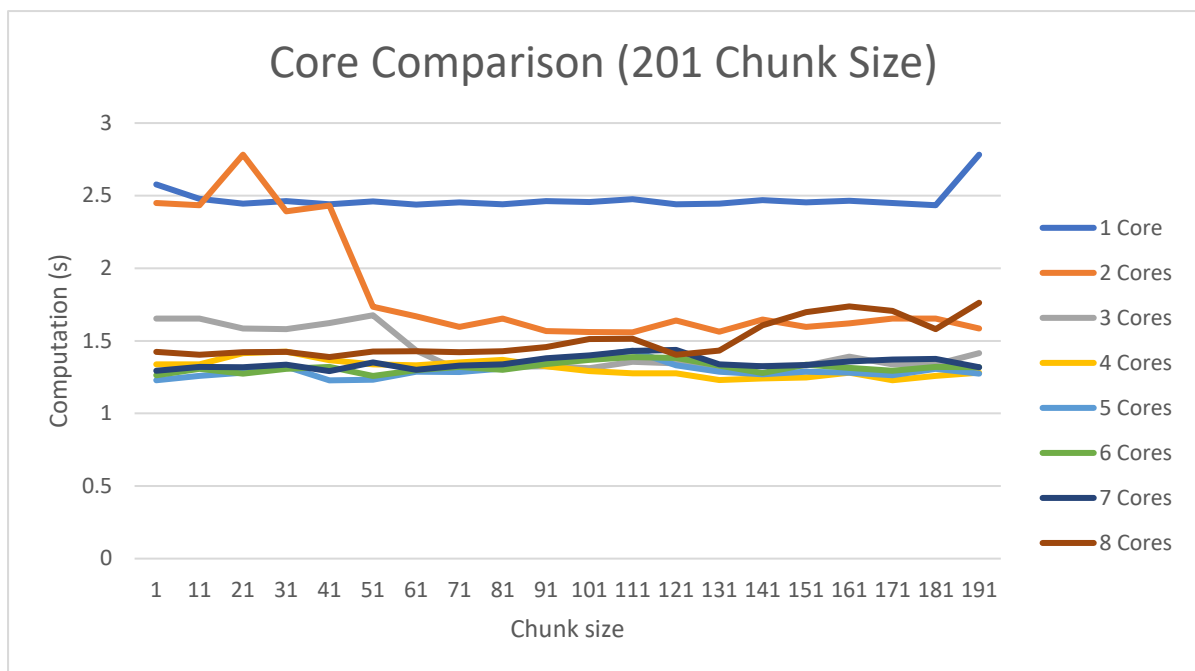


Speedup for chunksize

(See Computations.xlsx, Sheet: "Speedup")



(See Computations.xlsx, Sheet: "Core 201 chunk size")



Output from running algorithm

