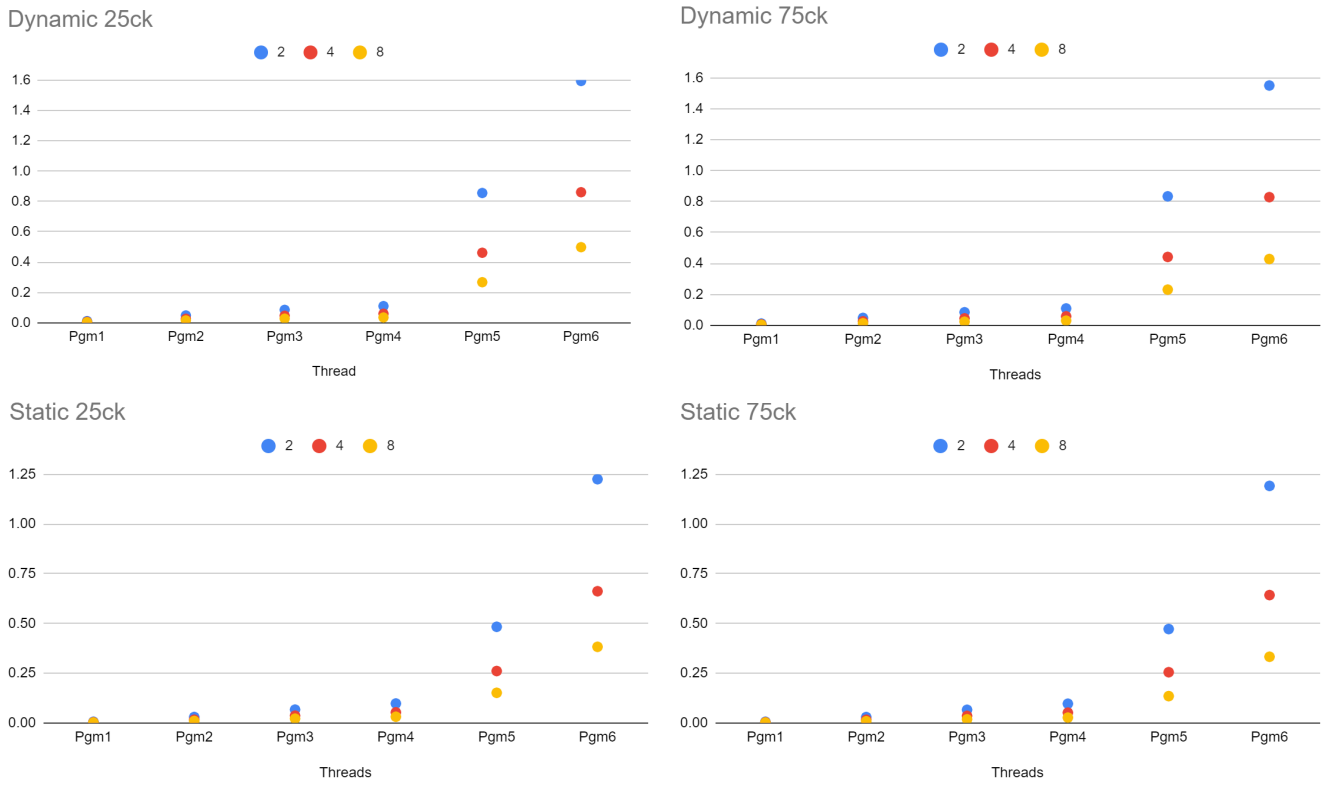
Henry Feng

Reporting Data

Graphic Data



The graphs above are separately graphed based on Dynamic vs Static and 25 chunks vs 75 chunks. I would like to highlight a few details with respect to the graph. Static did better than dynamic(the graphs are scaled differently) and 75 chunks did better than 25 chunks marginally.

Data Collection Method

The testes were done on each individual pgm file and each test took the average of 10 trails and the last trail was tested for correctness. Both static and dynamic were tested under the same loop, by clearing the output data every loop. Finally the image is dynamically allocated and deleted at the end of each iteration. The numerical values are below.

Numerical Values

Chunk 25 dynamic						
Thread	Pgm1	Pgm2	Pgm3	Pgm4	Pgm5	Pgm6
2	0.010806	0.047887	0.084888	0.109617	0.855367	1.59378
4	0.005801	0.025949	0.045965	0.059395	0.461348	0.860465
8	0.003482	0.015176	0.026889	0.034675	0.267143	0.498073
Chunk 75 dynamic						
Threads	Pgm1	Pgm2	Pgm3	Pgm4	Pgm5	Pgm6
2	0.010877	0.04739	0.083849	0.108519	0.833639	1.55039
4	0.005633	0.025059	0.044358	0.057711	0.441577	0.828434
8	0.003056	0.012963	0.022848	0.02972	0.230279	0.428275
Chunk 25 Static						
Threads	Pgm1	Pgm2	Pgm3	Pgm4	Pgm5	Pgm6
2	0.005429	0.029383	0.066455	0.097281	0.483086	1.22537
4	0.002941	0.015846	0.036055	0.052734	0.260603	0.661918
8	0.001761	0.009375	0.021072	0.030868	0.151081	0.38236
Chunk 75 Static						
Threads	Pgm1	Pgm2	Pgm3	Pgm4	Pgm5	Pgm6
2	0.005438	0.029162	0.065656	0.096215	0.471799	1.19165
4	0.002878	0.015554	0.034965	0.051323	0.254681	0.642594
8	0.001544	0.008048	0.017887	0.026428	0.134442	0.332488

Testing

Answers were allocated into an answer matrix similarly to the input. Used a double for loop to compare the results.

```
bool fail = false;
/*
for (int i = 0; i < 10; i++)
{
    for (int j = 0; j < 10; j++) {
        std::cout << "out: " << outputImage[i][j] << "answer: " << answerImage[i][j] << std::endl;
    }
}
*/
for (int i = 0; i < image_height; i++)
{
    for (int j = 0; j < image_width; j++) {
        if (outputImage[i][j] != answerImage[i][j]) {
            fail = true;
        }
    }
}
if (fail == false) {
    std::cout << "sucess" << std::endl;
}
```

Analysis

Static vs Dynamic

Static allocation did better than dynamic allocation due the equal sizing of the chunk size. Since dynamic allocation probably also incurs additional overhead, static allocation would do better in most iterations.

25ck vs 75ck

The chunk size had a marginal effect on the program. However, The chunk size of 75 did consistently better than the chunk size 25. This is because it allows the utilization of the threads more consistently if the chunk size is larger. During small pictures and low thread numbers it can be observed that chunk size 25 did better in certain situations.

Number of threads

Greater number of threads provided better performance especially during larger images. This effect is most prominently shown by picture 5 and 6.where the difference in time is nearly halved.

Conclusion

OpenMp dynamic should be used along with tasks that have a more variable timing scheme. Chunk size should be relatively large, but it can also hurt the performance if the size is too large. Overall the Static 75ck 8threads program did the best in terms of the current program.