CS3210 Lab 6

Aaron Sng (A0242334N)

17 November 2021

Machines Used

- soctf-pdc-001 - Intel Xeon Silver 4114 with 10 physical cores and 20 hyperthreads
- soctf-pdc-002 - Intel Xeon Silver 4114 with 10 physical cores and 20 hyperthreads
- soctf-pdc-010 - Intel i7-7700 with 4 physical cores and 8 hyperthreads
- soctf-pdc-011 - Intel i7-7700 with 4 physical cores and 8 hyperthreads

Exercise 4

	Time (seconds)
i7-7700	3.68
Xeon Silver 4114	5.65

Table 1: Sequential Runtime on i7 and Xeon Silver

Num Processes	2	4	6	8	10	16
i7-7700	2.02s	1.12s	1.31s	1.12s	1.20s	1.14s
Xeon Silver 4114	3.02s	1.61s	0.90s	1.15s	0.77s	0.82s

Table 2: Runtime on i7 and Xeon Silver using OpenMP

Num Processes	2	4	8	16
i7-7700	13.69s	18.79s	14.37s	12.67s
Xeon Silver 4114	18.83s	17.71s	17.47s	16.18s

Table 3: Runtime on i7 and Xeon Silver using MPI

Num Processes	2	4	6	8	10	16
i7-7700	1.82x	3.29x	2.81x	3.29x	3.07x	3.23x
Xeon Silver 4114	1.22x	2.29x	4.09x	3.2x	4.78x	4.49x

Table 4: Speedup on i7 and Xeon Silver using OpenMP

Num Processes	2	4	8	16
i7-7700	0.269x	0.196x	0.256x	0.290x
Xeon Silver 4114	0.195x	0.208x	0.211x	0.227x

Table 5: Speedup on i7 and Xeon Silver using MPI

The following observations can be made:

- On the OpenMP implementations, the speedup factor does not increase significantly after 10 processes for the Xeon Silver and 8 processes for the i7. The number coincides with the number of logical cores for the i7 and number of physical cores for the Xeon Silver. After surpassing this number of processes, it could be possible that the system Integer Sort begins to run in a roundrobin fashion, losing parallelism. As a result, further speedup was not observed after these numbers.
- The MPI implementation did not result in a speedup. In fact, the MPI implementation resulted in an overall slowdown of the system. It could be attributed to the communication overhead caused by having an MPI system. Communication overhead can be due to the point-to-point communication between processes, that maybe blocking functions. These functions overall creates a bottleneck, causing parallelism to be lost. As a result, the system becomes slower substantially.

Exercise 5

This script will be executed until the total number of physical cores in the system in steps of 2.

```
# Obtain the sequential time from reading from 'real' timing
export OMP_NUM_THREADS=1
time ./is.C.x # This time would be T1
# Obtain the time from reading from 'real' timing, P is the number of processors
    experimenting on
export OMP_NUM_THREADS=P
time ./is.C.x # This time would be T2
```

`speedup(p)` is obtained by taking $\frac{T1}{T2}$. Using Amdahl's law the following relationship can be obtained where *f*, the sequential portion, is expressed in terms of `p` and `speedup(p)`.

$$f = \frac{p}{p-1}(\frac{1}{speedup(p)} - \frac{1}{p})$$

# Processes	T1	T2	speedup(p)	f
2	27.242s	13.85s	1.97x	0.0165
4	27.242s	6.919s	3.93x	0.00531
6	27.242s	4.69s	5.81x	0.00659
8	27.242s	3.842s	7.091x	0.01831
10	27.242s	3.17s	8.59x	0.01818

Table 6: Amdahl's Law Verification on `soctf-pdc-001`

# Processes	T1	T2	speedup(p)	f
2	17.373s	8.99s	1.93x	0.0349
4	17.373s	4.721s	3.93x	0.0290

Table 7: Amdahl's Law Verification on `soctf-pdc-010`

f appears to vary significantly across machines. It is therefore hard to converge a certain value of *f*. Taking the average across all measurements, however, *f* appears to be 0.0184.