**Experiments of State Access Pattern:**

Experiment purpose:

1. Demonstating patterns work
2. Demonstrating the performance results are as predicted.

Experiment condition:

Cpu: Intel i7-7700HQ

Number of parallel: up to 4

Operation System: linux 4.15.0-generic

Compiler: g++ 7.3.0

FastFlow version: v.2.2

Workload: 100 transactions

Experiment:

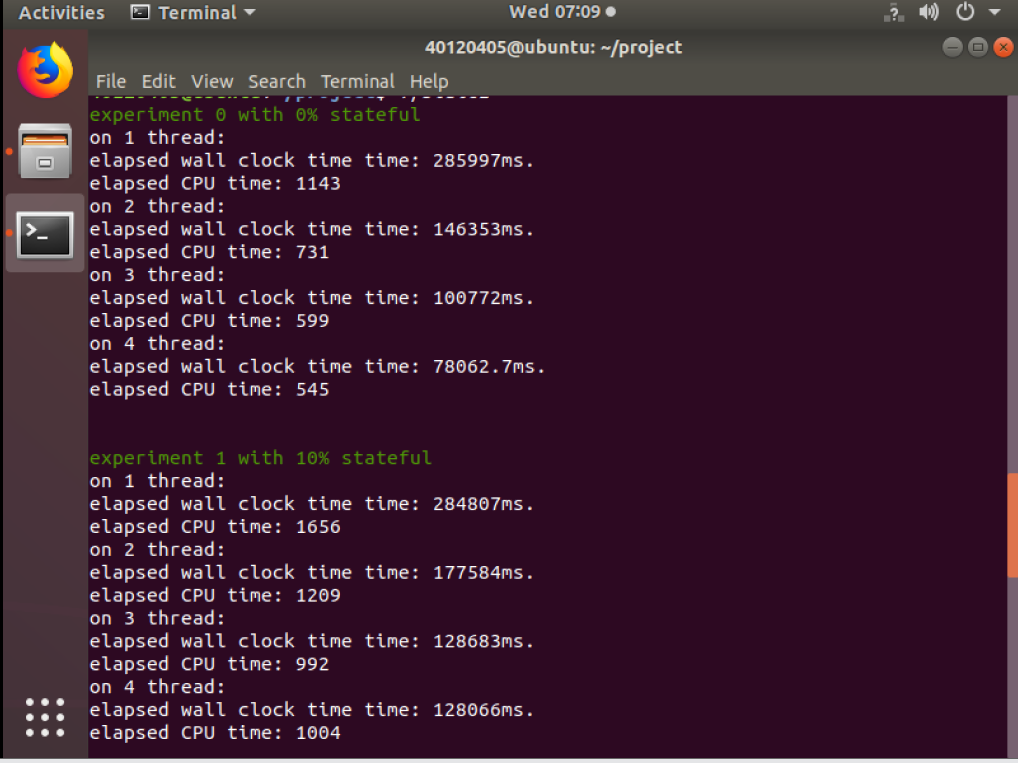
Experiment with diffident percentage of state access pattern (0%,10%,20%,40%,60%).

Execute the program with different parallelism degree (1,2,3,4)

**Experiment result:**

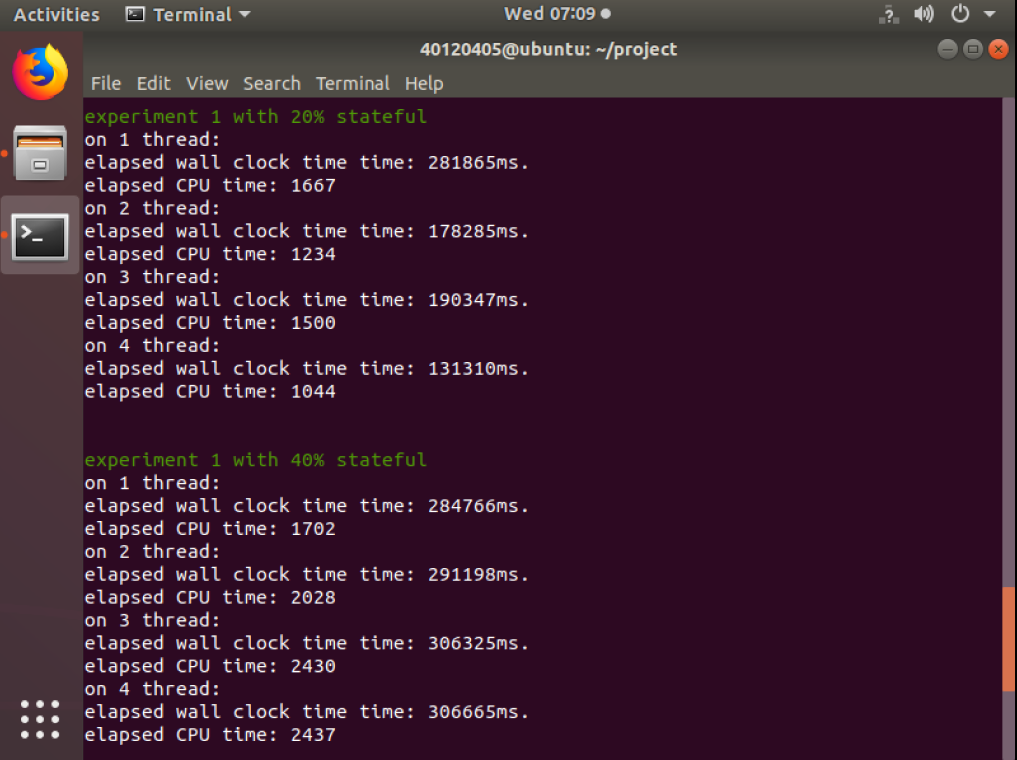
**Experiment with none state access pattern.**

**Experiment with 10% state access pattern.**

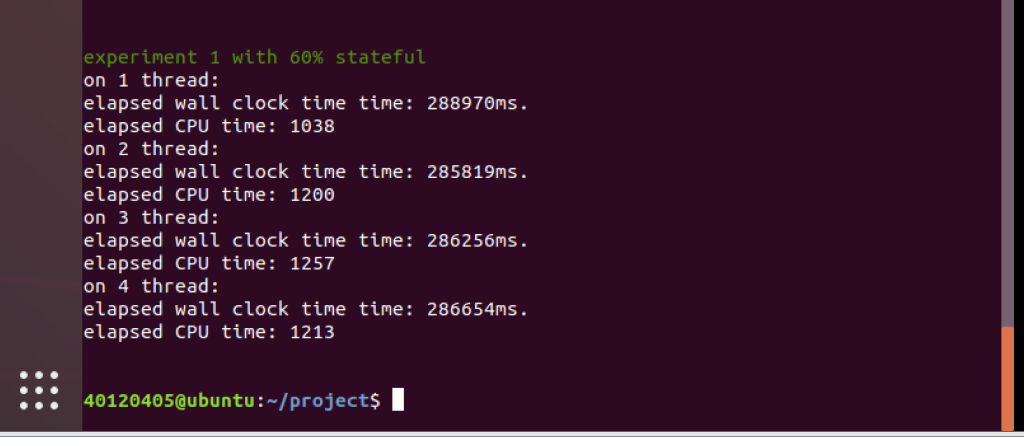


**Experiment with 20% state access pattern.**

**Experiment with 40% state access pattern:**



**Experiment with 60% state access pattern:**



**Wall clock time of processing:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Total number of workers: | 1 | 2 | 3 | 4 |
| P=0 | 286.0s | 146.4s | 100.8s | 78.1s |
| P=0.1 | 284.8s | 177.6s | 128.7s | 128.1s |
| P=0.2 | 281.9s | 178.3s | 190.3s | 131.3s |
| P=0.4 | 284.8s | 291.2s | 306.3s | 306.7s |
| P=0.6 | 288.9s | 285.8s | 286.3s | 286.7s |

**The calculated ideal parallel speedup:**

**We suppose all tasks running at 1 thread, the total process time is t.**

**Percent of serial tasks is p, so percent of parallel tasks is (1-p).**

**We also suppose the tasks will be run at n threads.**

**So if (1-p) parallel tasks run at n threads, and their total running time is:**

**(1-p)\*t/n + p\*t**

**Therefore we can get its parallel speedup:**

**t/[(1-p)\*t/n + p\*t] = 1/[(1-p)/n + p]**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Total number of workers: | 1 | 2 | 3 | 4 |
| P=0 | 1 | 2 | 3 | 4 |
| P=0.1 | 1 | 1.81 | 2.5 | 3.08 |
| P=0.2 | 1 | 1.67 | 2.13 | 2.5 |
| P=0.4 | 1 | 1.43 | 1.67 | 1.82 |
| P=0.6 | 1 | 1.25 | 1.37 | 1.43 |

**Actual Parallel speedup**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Total number of workers: | 1 | 2 | 3 | 4 |
| P=0 | 1 | 1.95 | 2.84 | 3.66 |
| P=0.1 | 1 | 1.60 | 2.21 | 2.22 |
| P=0.2 | 1 | 1.58 | 1.48 | 2.15 |
| P=0.4 | 1 | 0.98 | 0.93 | 0.93 |
| P=0.6 | 1 | 1.01 | 1.01 | 1.01 |

**Graph**

**Conclusion:**

From the result and graph above we can see the execute time and parallel speedup with different parallelism degrees, and its ideal speedup.

In the p=0, p=0.1 and p=0.2, the parallel speedup of each transactions are almost fit the ideal data.

But if the percentage of stateful pattern is high (p=0.4 and p=0.6), the multithread cannot offer any parallel speedup, which is not meet the standard situation. The reason maybe the communication time is too long, at the moment it’s very important to continue to find the root of bug.