1.

**open**

Good morning, Professors!

I am Tianbai, my final project is Adaptivity in Stateful Parallel Pattern. The paper was completed under the careful guidance and strong support of Peter. I am very grateful to all the professors for participating in my final presentation. Below I will give a brief introduction to the main contents of the paper, and I hope to get guidance from each professor.

2.

**outline**

I would like to talk about my project in five parts, firstly I will do an introduction to the project. Then I will explain in detail what is adaptivity and stateful access pattern, and next the fastflow parallel framework. Finally I will focus on my final project “Adaptivity in Stateful Parallel Pattern”, In order to understanding better, I will give an experiment in a paper.

3.

**introduction**

Nowadays, Processor technology has reached the bottleneck, and it’s difficult for standard single-threaded code to increase more the computing speed. Therefore, in the recent years, everyone turned their attention to parallel programming, and programming using a parallel programming framework is particularly important, which providing great convenience to developers. The popular mainstream parallel frameworks are TBB, OPENMP and MPI, etc [ˌɪt'setərə]., At the moment, stateful tasks account for a large proportion, However, in society the almost tasks we are processing at the moment by the parallel framework is still Stateless, that is, the internal state of the parallel task is not changed when the parallel framework is running, so the current mainstream parallel framework does not support stateful task. However, if you still need to perform stateful tasks in a parallel framework, the efficiency will be greatly reduced, and even the entire framework may be raising more errors than when dealing with stateless tasks. And in actual operation, the time spent on each task is different. The exact time required for each task depends on the task itself, and even base on the hardware environment at runtime. If you need to perform a lot of heavy tasks for a while,

you may need more threads, and instead you don't have to use so many threads when no many tasks, A good adaptivity algorithm may save lot’s of overhead in CPU, but at the moment still no person do the relevant experiment.Therefore, adaptive research on stateful tasks is the current top priority.

In this paper, we propose a good idea about implement an adaptive way with the stateful pattern and realize it for Fastflow programming model

4.

**Adaptivity**

Now I would like to explain what is adaptivity. Adaptivity is the process of automatically adjusting the processing method to obtain the best processing effect according to the data characteristics of the processed data during processing.

The number of workers will be automatically adjusted to make better use of the CPU.

5.

**stateful**

Now I would like to explain what is stateful. As patterns where the stage (in pipeline) or the worker (in farm) processes/threads should support internal state and support access to some more generalized notion of “pattern” global state.

Bank account transaction example: imitating the process of bank account transactions,

6.

**fastflow**

Now I would like to talk about fastflow, the parallel framework is what I used in this project.

.

Fastflow is a C++ high-efficiency parallel framework , which advocating high-level, pattern-based parallel programming.

Here you can see the overall structure of fastflow and some patterns inside

The performance of fastflow is very good. After a rigorous assessment, FastFlow is typically faster than some state-of-the-art parallel programming frameworks such as Intel TBB, OpenMP, Cilk in some cases

7.

In computer science, a design pattern is a general solution to a problem in programming. A design pattern provides a reusable architectural outline that may speed the development of many computer programs, parallel design pattern likes “Recipes“ to handle parallelism.

algorithm skeleton

Pre-defined parallel high-order functions provided as constructs or lib calls

8.

9.

Adaptivity in stateful pattern

Next I will talk my final project adaptivity in stateful pattern

The purpose is Through adding the adaptivity function to process the stateful tasks, to make sure the average consumption is always keep in the specific range .And we also observe the performance of adaptivity each different stateful percentage.

We do this adaptivity function by recoding the during time between recent 10 tasks, and we set the specific total workers and some workers active at the begin, then if the consume larger than specific period, we need the adaptivity, so the active workers add 1 during every 10 tasks, if the consume less than specific second, that means we don’t need the adaptive, so the active workers reduce 1 during every 10 task .

10.

11.

12.

**example**

Here I will tell a example how the program works, this example is an experiment from the paper, the adaptivity with 1% stateful tasks, here 1% means Stateful tasks account for 1% of the total task. In the upper half figures, X axis is the Elapsed time, or the ID of tasks, Y axis is the average consume of 10 tasks, the controlled experiment consumption range is from 2000ms to 5000ms per 10 tasks.And the second half shows Number of workers changes over the time.

In order to better carry out the experiment of adaptivity. I set first 50 tasks are normal tasks,which is in this part, which cost 4000 around per 10 tasks at 5 workers, 50~100 are hard tasks at this part which will take more time than the normal tasks, and 200~400 are easy tasks and takes less time in this part.

13

Here at the beginning the Emitter send all the tasks to 5 Workers

14.

After a short time, because the average consume still not more than 5000ms, so the number of active Workers keep at 5

15.

After task ID 50, the tasks will become hard task which takes more time, therefore the average consume would be over 5000ms, Emitter start send tasks to a new worker, and number of active Workers automatically increase 1

16.

After a short time,although the number of Workers has been increased, the average consumption still higher than 5000ms, so that counts of active Worker still keeps increase

17.

with the number of active Workers increase to 10, the process speed has got quickly enough, therefore the average consumption back to around 3000, then number of active Workers keep at 10

18.

after task ID 200, we instead all the tasks by the easy task, therefore the average consume would be less than 2000ms,we don’t need so many workers at this time ,emitter stop send tasks to a exist active worker, and number of active Workers automatically reduce 1

19.

with the number of active Workers reduce to 3, the process speed has got slowly enough, therefore the average consumption back to around 3000, then number of active Workers keep at 3 all the time

20.

**performance**

I think after the experiment , the implement is almost fit the requirement of Adaptivity, but I think the performance is still not very well with the high percentage stateful , that may be the reason that scheduler may need more time for scheduling the stateful tasks when using 16 or more threads, Future work will address the short-comings of our approach– it’s possible that the adaptivity may not be very well with the high percentage stateful, by focusing on the scheduling algorithm and using the real stream

21.

That’s all of my presentation, if you have any questions about my project, please no hesitate and ask me, thank you very much!