EI338 Computer System Engineering Homework 7

Zhou Litao 518030910407 F1803016

November 6, 2020, Fall Semester

Exercise 1 Provide three programming examples in which multithreading provides better performance than a single-threaded solution.

Solution.

1. In Matrix Multiplication, matrix can be tiled to execute block multiplication within different threads, so that the computation can be accelerated in parallel.
2. In services that are related to communication between users, multithreading can help the server handle multiple requests at the same time, so that one user's request won't be easily blocked by others if error occurs.
3. In modern AI applications, models that require massive computation such as neutral networks have taken multithreading into popular use, in order to make sure that multiple data can be processed at the same time to make the AI system efficient.

□

Exercise 2 Consider the following code segment:

```
pid_t pid;
pid = fork();
if (pid == 0) { /* child process */
     fork();
     thread_create(. . .);
}
fork();
```

1) How many unique processes are created?
2) How many unique threads are created?

Solution.

1. In addition to the main process, 5 new processes are created.
2. 2 new threads are created.

My experiment result can be found below. □

Exercise 3 The program shown in below uses the Pthreads API. What would be the output from the program at LINE C and LINE P?

Solution.

1. Line C outputs 5
2. Line P outputs 0

□

```c
int main(int argc, char *argv[]){
    pid_t pid;
    pthread_t tid;
    pthread_attr_t attr;

    pid = fork();

    if (pid == 0) {
        fork();
        pthread_attr_init(&attr);
        pthread_create(&tid,&attr,runner,NULL);
        // pthread_join(tid,NULL);
    }
    fork();
    sleep(5);
    printf("Create Process\n");
}

void *runner(void *param){
    printf("Create Thread %ld \n",pthread_self());
    pthread_exit(0);
}
```

输出 终端 调试控制台 问题 3 2: 任务 - Shell: opam_env

```
^C
root@8673101f53df:/GIT/2020Fall/EI338/Assignment# gcc try.c -o try -pthread
root@8673101f53df:/GIT/2020Fall/EI338/Assignment# ./try
Create Thread 140055770871552
Create Thread 140055770871552
Create Process
Create Process
Create Process
Create Process
Create Process
Create Process
root@8673101f53df:/GIT/2020Fall/EI338/Assignment# gcc try.c -o try -pthread
root@8673101f53df:/GIT/2020Fall/EI338/Assignment# ./try
Create Thread 140329536243456
Create Thread 140329536243456
Create Process
Create Process
Create Process
Create Process
Create Process
Create Process
root@8673101f53df:/GIT/2020Fall/EI338/Assignment#
```

```
#include <pthread.h>
#include <stdio.h>

int value = 0;
void *runner(void *param); /* the thread */

int main(int argc, char *argv[])
{
pid_t pid;
pthread_t tid;
pthread_attr_t attr;

   pid = fork();

   if (pid == 0) { /* child process */
      pthread_attr_init(&attr);
      pthread_create(&tid,&attr,runner,NULL);
      pthread_join(tid,NULL);
      printf("CHILD: value = %d",value); /* LINE C */
   }
   else if (pid > 0) { /* parent process */
      wait(NULL);
      printf("PARENT: value = %d",value); /* LINE P */
   }
}

void *runner(void *param) {
   value = 5;
   pthread_exit(0);
}
```