report_311512040

Parse program arguments

首先利用`getopt()`將終端傳入的指令依照不同的變數切開，arg[2]能知道命令有幾個threads，再另一個float還有兩個陣列分別去紀錄time_wait的值和各個thread的policy以及priority，這與後面設置thread的屬性有關。

```
char *optstring = "n:t:s:p:";
ch = getopt(argc, argv, optstring)
```

Create num_threads worker threads

在此步驟設置create thread會用到的一些屬性，並用陣列的方式儲存，pthread_t的作用代表thread的id，pthread_attr_t存取thread的屬性，pthread_info_t是一個struct存取各thread的資訊，sched_param也是一個struct，裡面可以設定priority，其中NUM_THREADS代表thread的數量。

```
pthread_t threads[NUM_THREADS];
pthread_attr_t attr[NUM_THREADS];
pthread_info_t thread_data_array[NUM_THREADS];
struct sched_param params[NUM_THREADS];
```

Set CPU affinity

將所有執行緒設定在同一個CPU上跑，這樣子schedule才有意義。

```
#define __USE_GNU
#include <sched.h> // for sched_getcpu()
int cpu_id = 3; // set thread to cpu3
cpu_set_t cpuset;
CPU_ZERO(&cpuset);
CPU_SET(cpu_id, &cpuset);
pthread_setaffinity_np(pthread_self(), sizeof(cpuset), &cpuset);
```

Set the attributes to each thread

利用`pthread_attr_init()`初始化線程屬性變量；利用`pthread_attr_setinheritsched()`設置線程繼承性；利用`pthread_attr_setschedpolicy()`設置線程調度策略，第二個參數是要放整數，所以必須先將終端傳入的NORMAL,FIFO轉為整數；利用`pthread_attr_setschedparam()`設置優先度，NORMAL不能給-1，因為sched_priority要在1～127的範圍。

```
pthread_attr_init(&attr[i]); /*初始化線程屬性變量*/
pthread_attr_setinheritsched(&attr[i],PTHREAD_EXPLICIT_SCHED); /*設置線程繼承性*/
pthread_attr_setschedpolicy(&attr[i],int_policies[i]);
pthread_attr_setschedparam(&attr[i], &params[i]);
```

Create Thread

設定好屬性後就能利用`pthread_create()`將thread建立，第一個參數為pthread_t *,第二個參數則是我前面設定好的pthread_attr_t, 第三個參數是我們要讓這個thread執行的程式，第四個則是該程式的參數。

```
pthread_create(&threads[i],&attr[i],thread_func,(void *)&thread_data_array[i]);
```

Start all threads at once

再main function內利用`pthread_barrier_init()`來去設置要等待幾個thread，數量設置為NUM_THREADS+1，再將`pthread_barrier_wait()`放在thread_func的for迴圈之前，等到所有threads都被創建完成後才一起執行。

```
pthread_barrier_t barrier;
pthread_barrier_init(&barrier, NULL , NUM_THREADS+1); /*需要等待n+1*/
pthread_barrier_wait(&barrier);
```

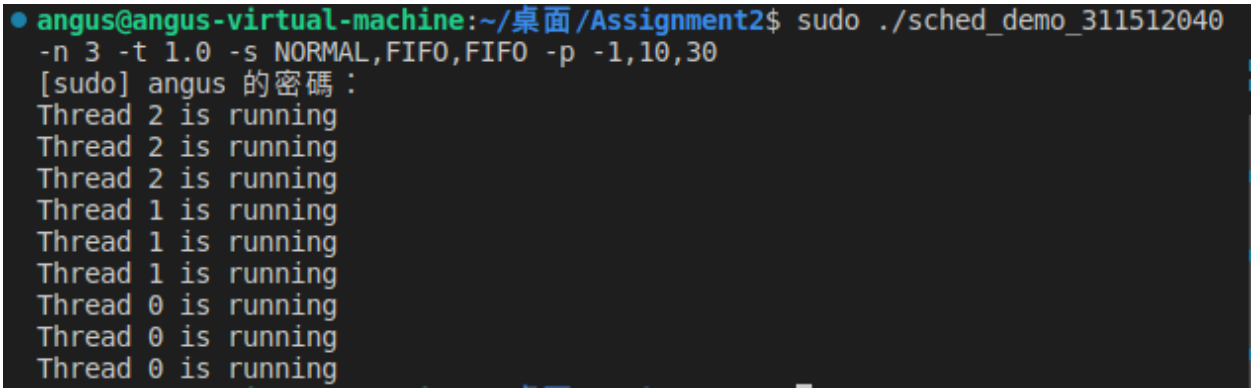
Busy for time_wait seconds

這部分是希望cpu值行一定的時間後能釋放資源然後重新schedule，利用`gettimeofday()`來去取得當前的時間，再計算當執行超過給定時間的t值後，釋放資源。

thread_info_t (thread的資訊)

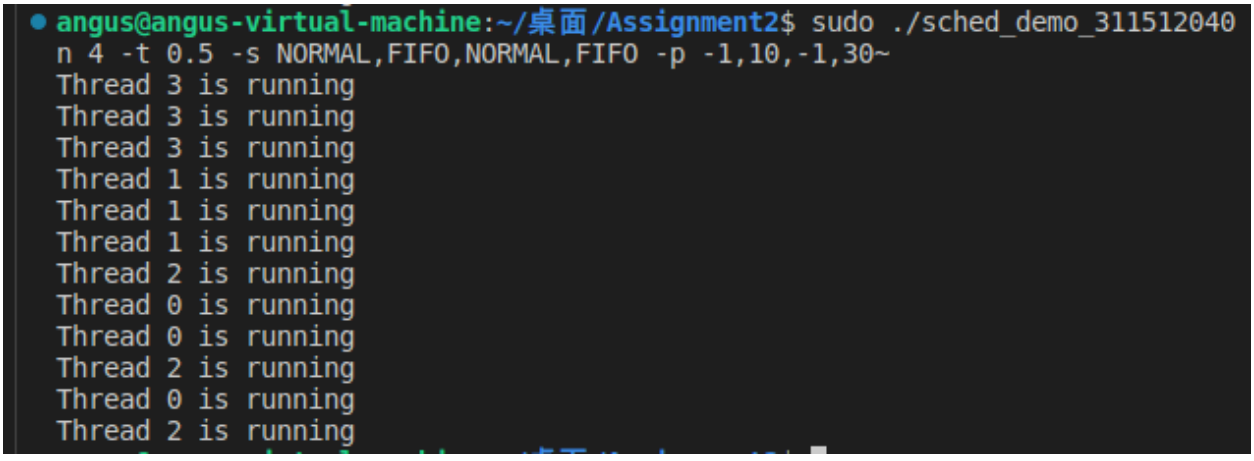
```
typedef struct {
    pthread_t thread_id;
    int thread_num;
    int sched_policy;
    int sched_priority;
    float time_wait;
}thread_info_t;
```

Describe the results of `./sched_demo -n 3 -t 1.0 -s NORMAL,FIFO,FIFO -p -1,10,3`



FIFO屬於real-time所以會比NORMAL優先，thread2的優先度又高於thread1所以順序優先為2 -> 1 -> 0

Describe the results of `./sched_demo -n 4 -t 0.5 -s NORMAL,FIFO,NORMAL,FIFO -p -1,10,-1,30`



因為thread3為FIFO且優先度最高，所以會先run完，接著是同為FIFO的thread1，接著thread2和thread0同為NORMAL沒有優先度的問題，所以會交錯執行。

Describe how did you implement n-second-busy-waiting

利用while迴圈不斷的去抓當前的時間，然後減去起始的時間來檢查是否已執行超過設定的時間。取得時間的函示利用`gettimeofday()`。

```
struct timeval start,end;
double starttime,endtime;
gettimeofday(&start,NULL);
starttime = (start.tv_sec*1000000+start.tv_usec)/1000000;
while (1)
{
gettimeofday(&end,NULL);
endtime = (end.tv_sec*1000000+end.tv_usec)/1000000;
if (endtime> starttime + my_data->time_wait)
break;
}
sched_yield();
```

▼ 參考資料

概念

作業系統筆記 之 CPU排程(CPU Scheduling)

創建 Thread

Pthreads 入門教程

pthread 簡要使用指南（五） 線程屬性(pthread_attr_t)

C語言pthread_create傳遞帶多個參數的函數& pthread_join

C/C++ Linux/Unix pthread 建立多執行緒用法與範例

第三個參數:void *(*func)(void *)總結

Parse program argument

getopt的用法與optarg

c++基礎之STL函數strtok以逗號和分號切割字符串為數字

指定CPU

C/C++ Linux/Unix 讓執行緒跑在指定 CPU 的方法 pthread_setaffinity_np

sched_param:描述調度參數的結構

struct sched_param 結構體

sched_param

多 Thread

重點參考 : **Posix多線程編程學習筆記**