并行程序设计报告

基准测试 单线程程序

```
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <time.h>
#include <semaphore.h>
#include <sys/time.h>
#include <unistd.h>
#include <sys/ipc.h>
#include <sys/sem.h>

#define MATRIX_LINE 2048
int matrix[MATRIX_LINE][MATRIX_LINE];
volatile int ans[MATRIX_LINE];


int main(int argc, char *argv[])
{
        //fill the matrix
        srand(time(NULL));
        int i, j;
        for (j = 0; j < MATRIX_LINE; j++)
        {

                for (i = 0; i < MATRIX_LINE; i++)
                {
                        int r = rand();
                        matrix[j][i] = r;
                }
        }

        struct timeval start;
        struct timezone tz;
        gettimeofday(&start, &tz);
        for (j = 0 ; j < MATRIX_LINE; j++)
                {
                        int ans_tmp;
                        for (i = 0; i < MATRIX_LINE; i++)
                        {
                                ans_tmp += matrix[j][i]*matrix[i][j];
                        }
                        ans[j] = ans_tmp;
                }
        struct timeval end;
        gettimeofday(&end, &tz);
        long int diff = end.tv_usec - start.tv_usec;
        printf("time used: %ld\n", diff);
```

```
        FILE *f;
        f = fopen("single.ans", "w");
        for (i = 0; i < MATRIX_LINE; i++)
        {
                fprintf(f, "%d\n", ans[i]);
        }
        fclose(f);
}
```

使用pthread并行计算的场合

注意编译时要加上 -pthread

```
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <time.h>
#include <semaphore.h>
#include <sys/time.h>
#include <unistd.h>
#include <sys/ipc.h>
#include <sys/sem.h>

#define MATRIX_LINE 2048
int matrix[MATRIX_LINE][MATRIX_LINE];
volatile int ans[MATRIX_LINE];


void *worker(void *arg)
{
        int i, j;
        if (*(int *)arg == 1)
        {
                for (j = 0; j < MATRIX_LINE / 2; j++)
                {
                        int ans_tmp;
                        for (i = 0; i < MATRIX_LINE; i++)
                        {
                                ans_tmp += matrix[j][i]*matrix[i][j];
                        }
                        ans[j] = ans_tmp;
                }
        }
        else if (*(int *)arg == 2)
        {
                for (j = MATRIX_LINE / 2; j < MATRIX_LINE; j++)
                {
                        int ans_tmp;
                        for (i = 0; i < MATRIX_LINE; i++)
                        {
                                ans_tmp += matrix[j][i]*matrix[i][j];
```

```
                        }
                        ans[j] = ans_tmp;
                }
        }

        return NULL;
}

int main(int argc, char *argv[])
{
        //fill the matrix
        srand(time(NULL));
        int i, j;
        for (j = 0; j < MATRIX_LINE; j++)
        {

                for (i = 0; i < MATRIX_LINE; i++)
                {
                        int r = rand();
                        matrix[j][i] = r;
                }
        }

        pthread_t p1, p2;

        struct timeval start;
        struct timezone tz;
        gettimeofday(&start, &tz);
        int arg1 = 1;
        int arg2 = 2;
        pthread_create(&p1, NULL, worker, (void *)&arg1);
        pthread_create(&p2, NULL, worker, (void *)&arg2);
        pthread_join(p1, NULL);
        pthread_join(p2, NULL);
        struct timeval end;
        gettimeofday(&end, &tz);
        long int diff = end.tv_usec - start.tv_usec;
        printf("time used: %ld\n", diff);
        FILE *f;
        f = fopen("pthread.ans", "w");
        for (i = 0; i < MATRIX_LINE; i++)
        {
                fprintf(f, "%d\n", ans[i]);
        }
        fclose(f);
}
```

使用fork方法并行计算的场合

fork方法使用共享变量需要使用shmget和shmat

```
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <time.h>
#include <semaphore.h>
#include <sys/time.h>
#include <unistd.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/types.h>
#include <sys/prctl.h>
#include <signal.h>
#include <errno.h>
#include <string.h>
#include <fcntl.h>
#include <sys/wait.h>
#include <malloc.h>
#include <stdarg.h>
#include <sys/types.h>
#include <sys/shm.h>
#define MATRIX_LINE 2048
int matrix[MATRIX_LINE][MATRIX_LINE];
volatile int ans[MATRIX_LINE];

sem_t son, son1;

static void sig_child(int signo);

int main(int argc, char *argv[])
{
        int shmid;
        int *shmaddr;
        signal(SIGCHLD, sig_child);
        shmid = shmget(IPC_PRIVATE, sizeof(int) * MATRIX_LINE, IPC_CREAT | 0600);
        sem_init(&son, 0, 0);
        sem_init(&son1, 0, 0);
        //fill the matrix
        srand(time(NULL));
        int i, j, i2, j2;
        for (j = 0; j < MATRIX_LINE; j++)
        {

                for (i = 0; i < MATRIX_LINE; i++)
                {
                        int r = rand();
                        matrix[j][i] = r;
                }
        }

        struct timeval start;
        struct timezone tz;
        gettimeofday(&start, &tz);
        int pid = fork();
```

```
        if (pid == 0)
        {
                shmaddr = (int *)shmat(shmid, NULL, 0);
                prctl(PR_SET_PDEATHSIG, SIGHUP);
                for (j = MATRIX_LINE / 2; j < MATRIX_LINE; j++)
                {
                        int ans_tmp;
                        for (i = 0; i < MATRIX_LINE; i++)
                        {
                                ans_tmp += matrix[j][i] * matrix[i][j];
                        }
                        shmaddr[j] = ans_tmp;
                }
                sem_post(&son);
                printf("son posted\n");
                return 0;
        }
        else if (pid > 0)
        {
                pid = fork();
                if (pid == 0)
                {
                        shmaddr = (int *)shmat(shmid, NULL, 0);

                        prctl(PR_SET_PDEATHSIG, SIGHUP);
                        for (j = 0; j < MATRIX_LINE / 2; j++)
                        {
                                int ans_tmp;
                                for (i = 0; i < MATRIX_LINE; i++)
                                {
                                        ans_tmp += matrix[j][i] * matrix[i][j];
                                }
                                shmaddr[j] = ans_tmp;
                        }
                        printf("son1 posted\n");
                        sem_post(&son1);
                        return 0;
                }
                else
                {

                        printf("main waitng\n");
                        sem_wait(&son);
                        sem_wait(&son1);
                        printf("main go\n");
                        struct timeval end;
                        gettimeofday(&end, &tz);
                        long int diff = end.tv_usec - start.tv_usec;
                        printf("time used: %ld\n", diff);
                }
        }
        else
        {
                printf("fork fail\n");
```

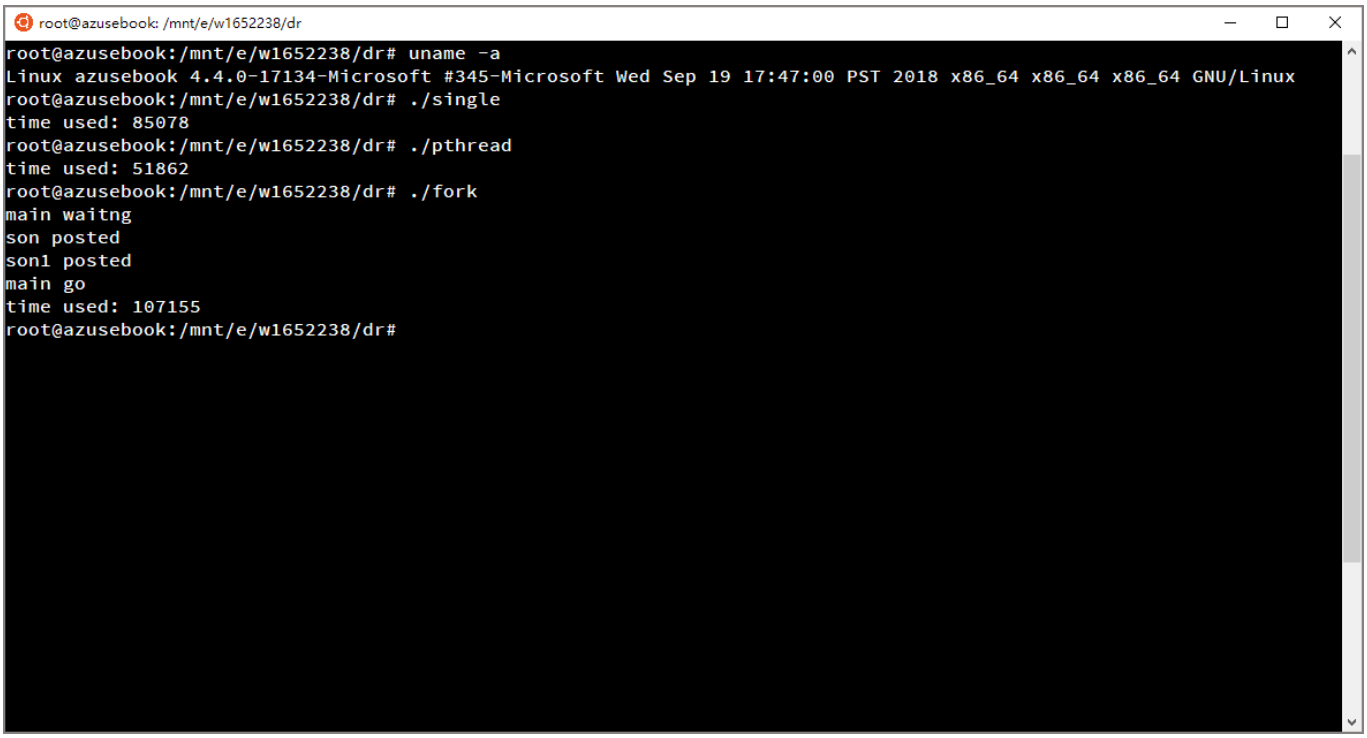
```
        }
        shmaddr = (int *)shmat(shmid, NULL, 0);
        FILE *f;
        f = fopen("fork.ans", "w");
        for (i = 0; i < MATRIX_LINE; i++)
        {
                fprintf(f, "%d\n", shmaddr[i]);
        }
        fclose(f);

        return 0;
}

static void sig_child(int signo)
{
        pid_t pid;
        int stat;
        while ((pid = waitpid(-1, &stat, WNOHANG)) > 0)
        {
                //printf("child %d exited with signal %d\n", pid, signo);
                fflush(stdout);
        }
}
```

makefile文件

```
# makefile

all : pthread fork single

single : single.c
        gcc -o single single.c

pthread :       pthread.c
        gcc pthread.c  -o pthread -pthread

fork    :       fork.c
        gcc fork.c -o fork -pthread

.PHONY: clean

clean:
        rm -f *.o pthread fork
```

运行结果

在windows subsystem linux上运行，可以看到运算pthread比单线程快，而fork反而更慢了


root@azusebook: /mnt/e/w1652238/dr
root@azusebook:/mnt/e/w1652238/dr# uname -a
Linux azusebook 4.4.0-17134-Microsoft #345-Microsoft Wed Sep 19 17:47:00 PST 2018 x86_64 x86_64 x86_64 GNU/Linux
root@azusebook:/mnt/e/w1652238/dr# ./single
time used: 85078
root@azusebook:/mnt/e/w1652238/dr# ./pthread
time used: 51862
root@azusebook:/mnt/e/w1652238/dr# ./fork
main waitng
son posted
son1 posted
main go
time used: 107155
root@azusebook:/mnt/e/w1652238/dr#