组会报告

徐益

2018 年 12 月 10 日

1 工作内容

1. 多线程系统优化；
2. 准备开题报告。

2 多线程系统优化

2.1 关闭超线程

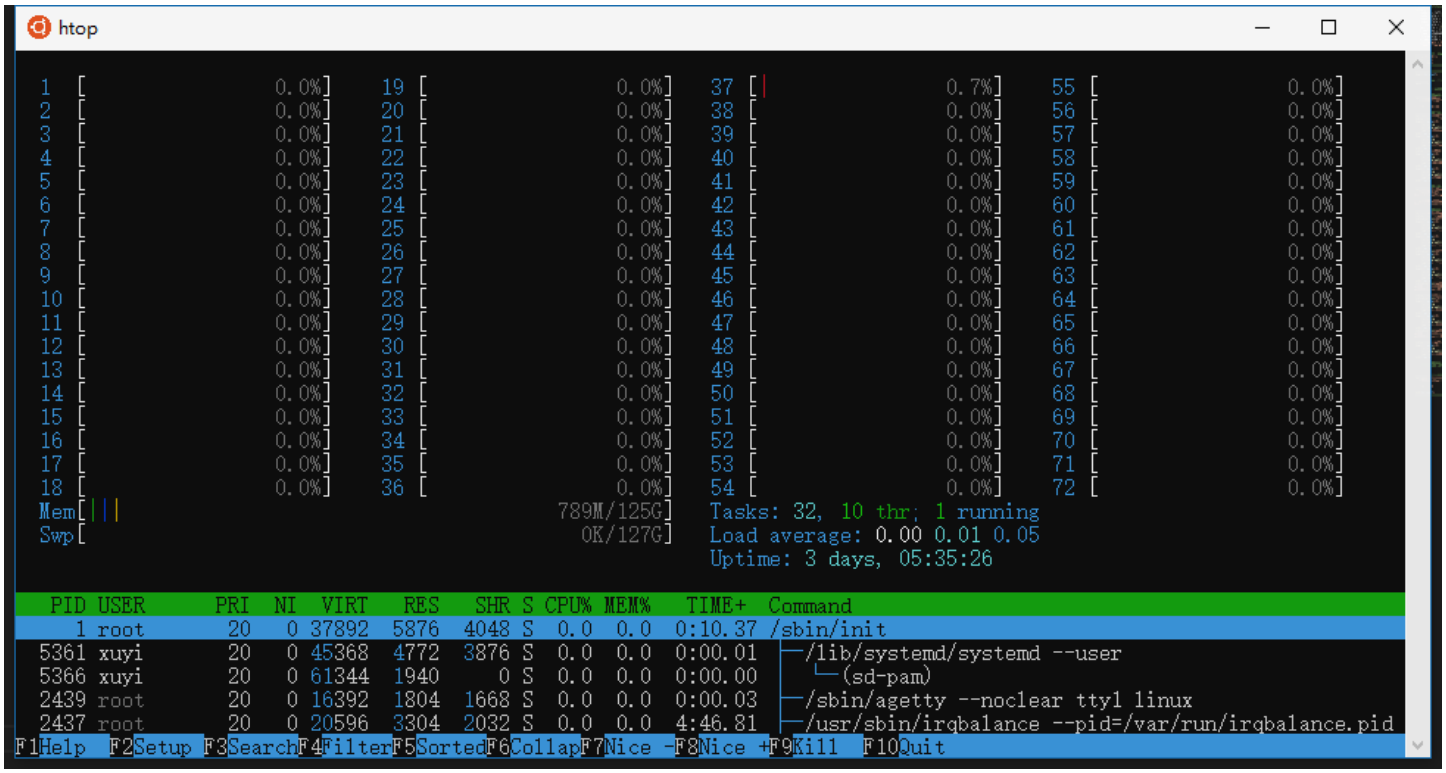

图 1: 关闭超线程后

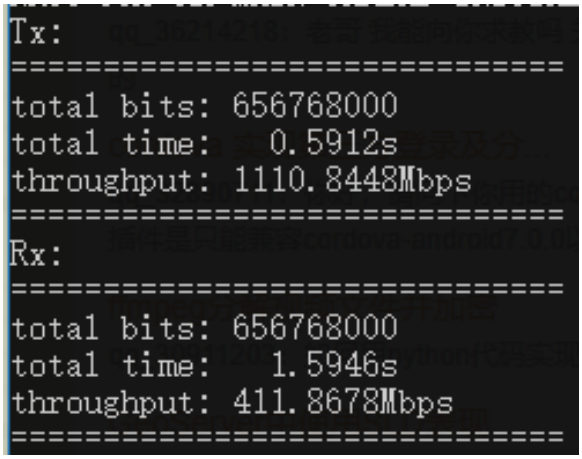

图 2: 吞吐量

2.2 线程绑定 CPU

```
typedef struct nr5g_phy_tx_sche_thrd_t
{
    throughput_counter_t *tp_counter;
    int32_t max_tb_byte_len; // 每个TB的最大长度 (Byte)
    int32_t pool_indx;

    int8_t **tx_data_buff;           // Tx数据Buffer
    nr5g_phy_tx_coef_t *tx_coef_buff; // Tx参数Buffer
    int32_t tx_buff_size;            // Tx中Buffer长度

    cplxf **tx_dpdk_data_buff; // Tx端DPDK数据Buffer
    int32_t tx_dpdk_buff_size; // Tx中Buffer长度

    sem_t *input_read_sem;   // Input读信号量
    sem_t *input_write_sem;  // Input写信号量
    sem_t *output_read_sem;  // Output读信号量
    sem_t *output_write_sem; // Output写信号量
    sem_t *destroy_sem;      // 线程销毁信号量
#ifdef SET_SETAFFINITY
    cpu_set_t *cpu_mask;
#endif
} nr5g_phy_tx_sche_thrd_t;
```

图 3: 方案

```
➜  tb_multi_thrd ./main
NR5G PHY TX MAIN THREAD START INITIALIZING...
NR5G PHY RX MAIN THREAD START INITIALIZING...
NR5G PHY TX MAIN THREAD INITIALIZING DOWN.
[1]    6401 segmentation fault (core dumped)  ./main
```

图 4: 错误

```
    pool_init(0, 1, 0);      // Tx Scheduling Thread Pool
    pool_init(1, 8, 1);      // Tx Thread Pool
#ifdef WIHT_CHANNEL
    pool_init(9, 1, 2);      // Channel Thread Pool
#endif
    pool_init(10, 1, 3);     // Rx Scheduling Thread Pool
    pool_init(11, 20, 4);    // Rx Thread Pool
```

图 5: 可能原因

```
void pool_init(int coreId_start, int _threadNum, int pool_index)
{
    int j = pool_index;
    struct pool_arg_t *pool_arg = (struct pool_arg_t*) malloc(sizeof(struct pool_arg_t) * 72);

    pool[j] = (struct Thread_Pool*) malloc(sizeof(struct Thread_Pool));
    assert(pool[j] != NULL);

    pthread_mutex_init(&(pool[j] -> mutex), NULL);
    pthread_cond_init(&(pool[j] -> cond), NULL);
    pool[j] -> taskHead = NULL;
    pool[j] -> isClose = false;
    pool[j] -> threadNum = _threadNum;
    pool[j] -> threadId = (pthread_t *) malloc(sizeof(pthread_t) * pool[j] -> threadNum);

    int i;
    //int coreId_start = 0;
    for(i = 0; i < pool[j] -> threadNum; ++i)
    {
        coreId[i] = coreId_start + i;
        pool_arg[i].coreId = coreId[i];
        pool_arg[i].pool_index = pool_index;
        if(pthread_create(&(pool[j] -> threadId[i]), NULL, thread_run, (void *)&pool_arg[i]))
        {
            printf("pthread_creat failed!\n");
            return;
        }
    }
}
```

图 6: pool_init

3 准备开题报告