知道了呀~

~ o(*￣▽￣*)o~

Linux将线程绑定到CPU内核运行

先介绍三个函数

一、pthread_setafftinity_np

在Linux上，我们可以使用pthread特定的pthread_setafftinity_np函数。通过设置其亲和性将每个线程固定到单个CPU

```
  //设置CPU亲和度，为0表示设置成功，非0值表示失败
              int rc
=pthread_setaffinity_np(p[i].native_handle(),sizeof(cpu_set_t),
&cpuset);
```

第一个参数是线程的句柄，第二个参数是CPU集合的大小，第三个参数是CPU集合的地址

线程和内核绑定成功返回值为0，失败返回一个非0值

二、sysconf(_SC_NPROCESSORS_CONF);

```
//获取电脑CPU内核的数量

int cpu_num;
cpu_num = sysconf(_SC_NPROCESSORS_CONF);
cout<<"cpu_num="<<cpu_num<<endl;
```

三、cpu_set_t结构体

公告

昵称： 知道了呀~
园龄： 3年8个月
粉丝： 71
关注： 21
+加关注

< 2022年5月 >

日	一	二	三	四	五	六
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

搜索

找找看

谷歌搜索

最新随笔

1.优化多线程安全的内存池

2.N叉树的前序遍历

3.求32位整数含1的个数

cpu_set_t这个结构体了。这个结构体的理解类似于select中的fd_set，可以理解为cpu集，也是通过约定好的宏来进行清除、设置以及判断：

有下列几种操作

```
//初始化，设为空
void CPU_ZERO (cpu_set_t *set);
//将某个cpu加入cpu集中
void CPU_SET (int cpu, cpu_set_t *set);
//将某个cpu从cpu集中移出
void CPU_CLR (int cpu, cpu_set_t *set);
//判断某个cpu是否已在cpu集中设置了
int CPU_ISSET (int cpu, const cpu_set_t *set);
```

完整代码

```
#include <stdio.h>
#include <iostream>
#include <thread>
#include <pthread.h>
#include <mutex>
#include <unistd.h>
 using namespace std;
 mutex _mutex;
 void func(int id,int cnt)
 {
        {
                lock_guard<mutex> lg(_mutex);
                cout<< "Thread id=  " << id << ":running on CPU "
<<sched_getcpu() << endl;
        }
    while (1) {

    }
}

int main()
{
        int cpu_num;
           cpu_num = sysconf(_SC_NPROCESSORS_CONF);
           cout<<"cpu_num="<<cpu_num<<endl;
        thread p[4];
        for(int i=0;i<4;i++){


                p[i]=thread(func,i,10);
                //cpu_set_t这个结构体了。这个结构体的理解类似于select中的
fd_set，可以理解为cpu集，也是通过约定好的宏来进行清除、设置以及判断：
                cpu_set_t cpuset;
                //初始化，设为空
                CPU_ZERO(&cpuset);
                //将某个cpu加入cpu集中
```

4.调整数组顺序，要求所有奇数在偶数前面，常数时间复杂度

5.字符串减法（大数）

6.C++11 实现信号量（吃水果问题）

7.文件系统--inode节点

8.mmap文件映射过程

9.Linux文件系统（上）

10.字符串加法

积分与排名

积分 - 270078

排名 - 3050

随笔分类

2018"双基"杯(3)

AC自动机(4)

C++数据结构(54)

DFS/BFS(39)

DP(36)

leetcode(12)

MATLAB教程(1)

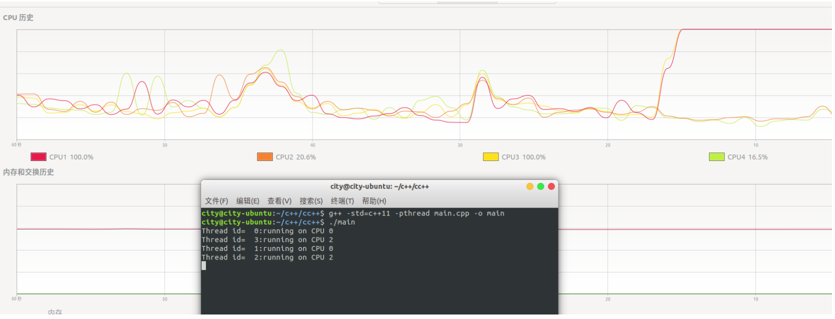
PTA(7)

Python(5)

python 网络爬虫(3)

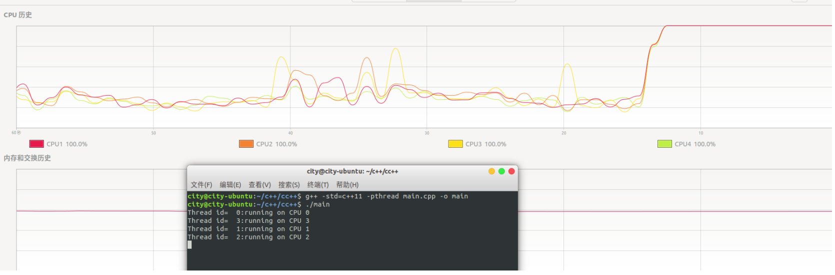
```
                CPU_SET(i,&cpuset);

                //设置CPU亲和度，为0表示设置成功，非0值表示失败
                int rc
=pthread_setaffinity_np(p[i].native_handle(),sizeof(cpu_set_t),
&cpuset);
                if(rc!=0){
                        cout<<"SET  CPU ERROR!\n";
                }


        }
        for(int i=0;i<4;i++){
                p[i].join();
        }
        return 0;

}
```

将4个线程绑定在第一个和第三个CPU内核运行



```
city@city-ubuntu:~/c++/cc++$ g++ -std=c++11 -pthread main.cpp -o main
city@city-ubuntu:~/c++/cc++$ ./main
Thread id=  0:running on CPU 0
Thread id=  3:running on CPU 2
Thread id=  1:running on CPU 0
Thread id=  2:running on CPU 2
```

将4个线程绑定在各自对应的4个CPU内核运行



```
city@city-ubuntu:~/c++/cc++$ g++ -std=c++11 -pthread main.cpp -o main
city@city-ubuntu:~/c++/cc++$ ./main
Thread id=  0:running on CPU 0
Thread id=  3:running on CPU 3
Thread id=  1:running on CPU 1
Thread id=  2:running on CPU 2
```

等风起的那一天，我已准备好一切

分类: C++数据结构, 计算机网络基础

好文要顶 关注我 收藏该文

知道了呀~
关注 - 21
粉丝 - 71

+加关注

1 0

shell编程(5)

STL(20)

并查集(6)

博弈论(2)

操作系统(19)

更多

随笔档案

2020年12月(1)

2020年11月(1)

2020年9月(1)

2020年8月(24)

2020年7月(16)

2020年6月(11)

2020年5月(25)

2020年4月(34)

2020年3月(26)

2020年2月(32)

2020年1月(3)

2019年12月(5)

2019年11月(14)

2019年10月(11)

2019年9月(30)

« 上一篇： Windows多线程与线程绑定CPU内核
» 下一篇： 柱状图的最大矩形--单调栈

posted @ 2020-07-28 10:11 知道了呀~ 阅读(1520) 评论(0) 编辑 收藏 举报

刷新评论 刷新页面 返回顶部

登录后才能查看或发表评论，立即 登录 或者 逛逛 博客园首页

编辑推荐：
· 踩到一个关于分布式锁的非比寻常的BUG！
· 通过代码解释什么是API，什么是SDK？
· 记一次生产问题的排查，让我领略了算法的重要性
· 阴影进阶，实现更加的立体的阴影效果！
· 如何写好B端产品的技术方案？

OpenHarmony 开发板免费试用活动
搭载人工智能应用处理器，价值千元！
报名参与互动还有机会赢取神秘奖品
立即申请
HUAWEI DEVELOPERS

最新新闻：
· 微软、苹果和谷歌将支持FIDO无密码登录
· IT老兵聊如何成为Sun公司第8号员工，网友：怀念那个“从无到有”的技术年代
· 亚马逊AWS为什么不分拆？
· 58同城杀入VC
· 2021 年度Go开发者调查
» 更多新闻...

历史上的今天：
2019-07-28 线段树（模板）
2019-07-28 Easy Climb UVA - 12170 滚动dp +离散化+ 单调队列优化
2019-07-28 POJ 2823 滑动窗口 单调队列模板

更多

递归

母函数

汉诺塔

阅读排行榜

1. 哈夫曼编码（理解）(80154)
2. 满二叉树、完全二叉树、平衡二叉树、最优二叉树(57422)
3. Python 数据的输入(12583)
4. 费马小定理(9493)
5. 蓝桥杯： 基础练习 十六进制转八进制(7043)

评论排行榜

1. 满二叉树、完全二叉树、平衡二叉树、最优二叉树(9)
2. 蓝桥杯： 基础练习 十六进制转八进制(9)
3. C++类的内存分布(4)
4. C++内存池的实现和原理（二）(2)
5. [蓝桥杯2016初赛]卡片换位 BFS(2)

推荐排行榜

1. 满二叉树、完全二叉树、平衡二叉树、最优二叉树(9)
2. 蓝桥杯： 基础练习 十六进制转八进制(6)
3. 哈夫曼编码（理解）(5)

4. C++迭代器失效问题（insert、erase）(4)

5. 乘法逆元(4)