Linux进程、线程和调度(3)

讲解时间：9月13日、9月15日、9月19日、9月22日晚20点

宋宝华 <21cnbao@gmail.com>

报名录播：

http://edu.csdn.net/course/detail/5995

扫描二维码报名

麦当劳喜欢您来，喜欢您再来

扫描关注

Linuxer

9.19第三次课大纲

1. CPU/IO消耗型进程
2. 吞吐率 vs. 响应
3. SCHED_FIFO、SCHED_RR
4. SCHED_NORMAL和CFS
5. nice、renice
6. chrt

练习题

1. 运行2个高CPU利用率进程，调整他们的nice
2. 用chrt把一个死循环进程调整为SCHED_FIFO
3. 阅读ARM的big.LITTLE架构资料，并论述为什么ARM要这么做？

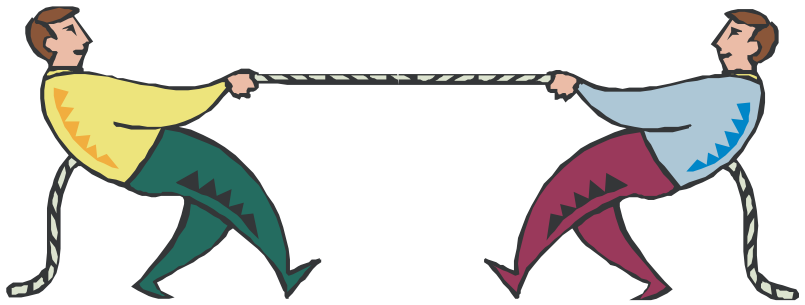
吞吐 vs. 响应

- 吞吐和响应之间的矛盾

✓ 响应：最小化某个任务的响应时间，哪怕牺牲其他的任务为代价

✓ 吞吐: 全局视野，整个系统的workload被最大化处理

吞吐



响应

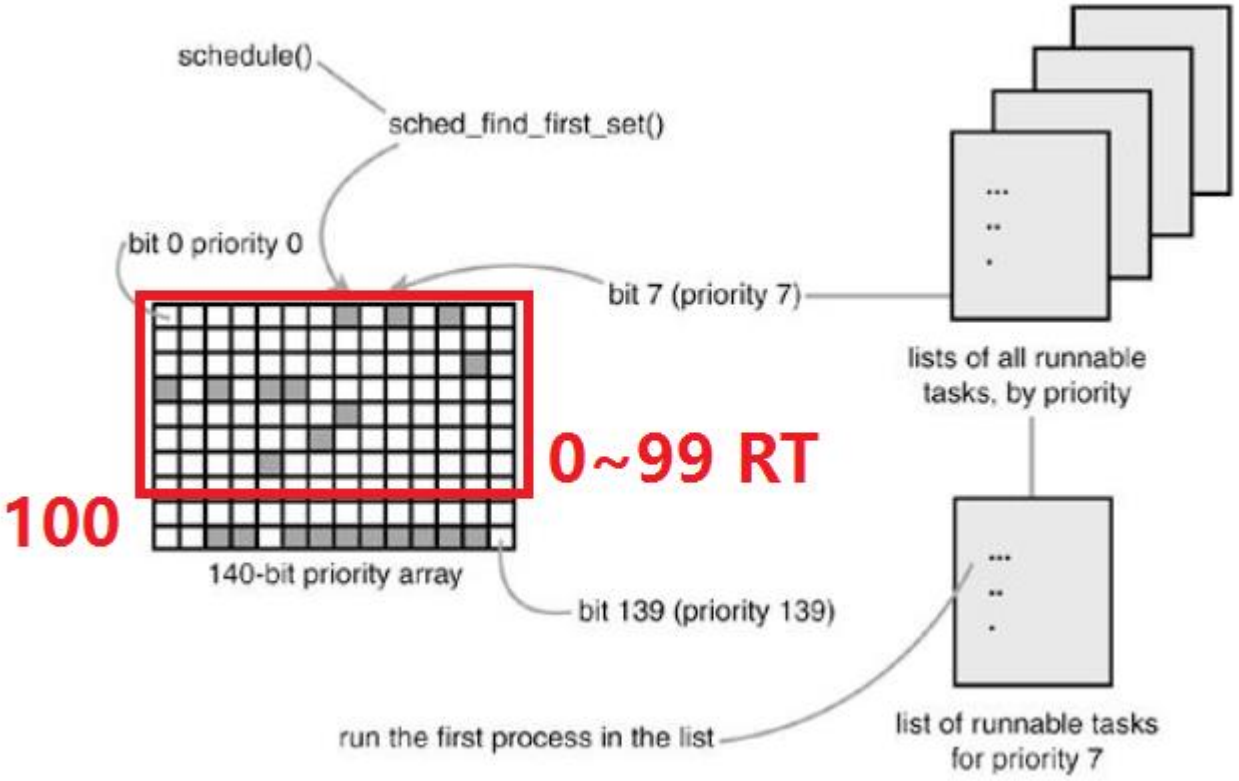
I/O消耗型vs. CPU消耗型

- IO bound: CPU利用率低，进程的运行效率主要受限于I/O速度；
- CPU bound: 多数时间花在CPU上面(做运算).

big.LITTLE

早期2.6:优先级数组和Bitmaps

- 0~139
- 某个优先级有TASK_RUNNING进程, 响应bit设置1。
- 调度第一个bitmap设置为1的进程



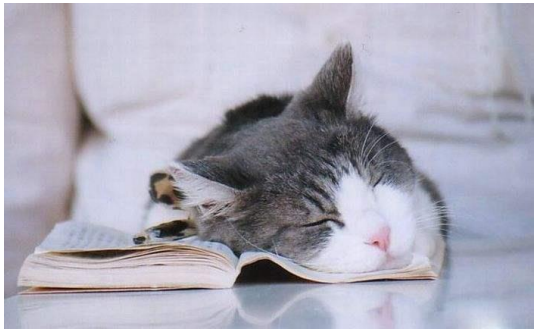
实时进程调度

- SCHED_FIFO：不同优先级按照优先级高的先跑到睡眠，优先级低的再跑；同等优先级先进先出。
- SCHED_RR：不同优先级按照优先级高的先跑到睡眠，优先级低的再跑；同等优先级轮转。

早期2.6:非实时进程的调度和动态优先级

- 在不同优先级轮转
- -20 ~ +19的nice值
- 根据睡眠情况，动态奖励和惩罚

惩罚!!



奖励

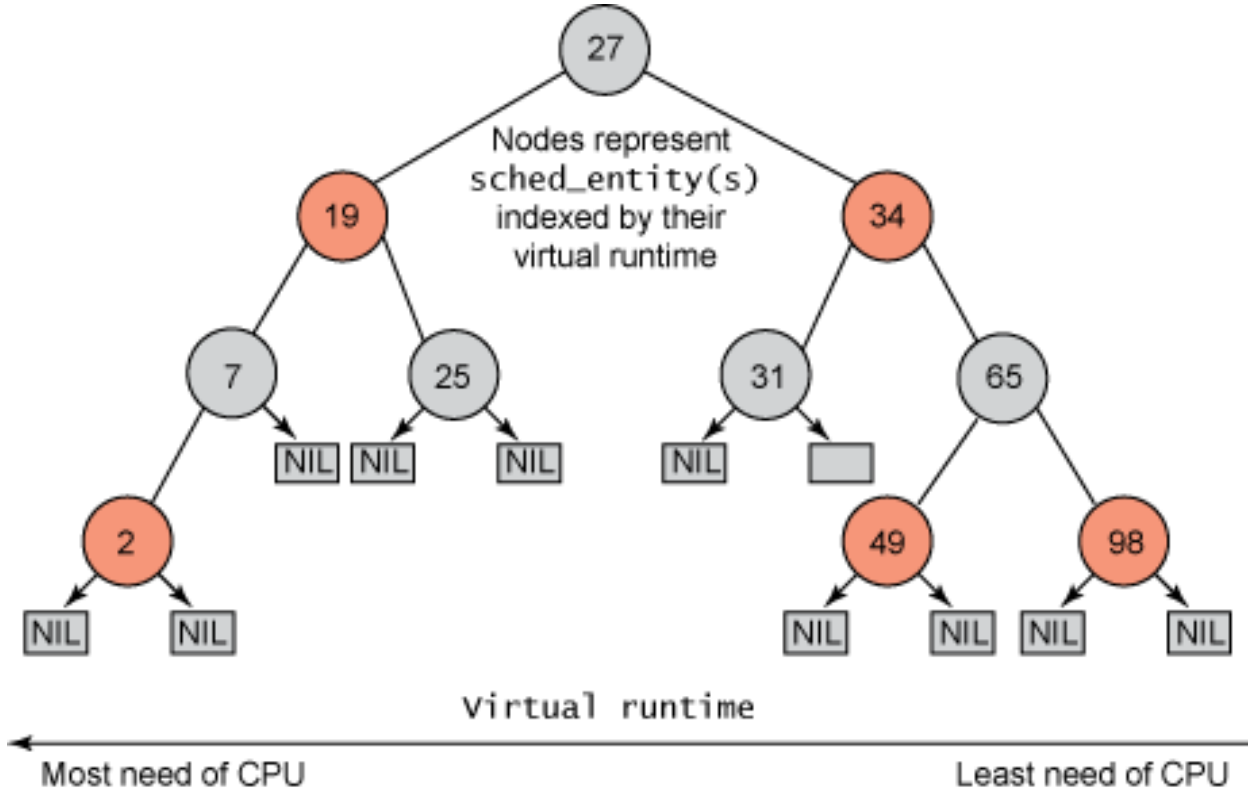
rt的门限

在period的时间里RT最多只能跑runtime的时间

/proc/sys/kernel/sched_rt_period_us
/proc/sys/kernel/sched_rt_runtime_us

CFS：完全公平调度

红黑树，左边节点小于右边节点的值
运行到目前为止vruntime最小的进程
同时考虑了CPU/IO和nice



CFS weight

vruntime += delta NICE_0_LOAD/ se.weight;*

```
static const int prio_to_weight[40] = {
 /* -20 */     88761,     71755,     56483,     46273,     36291,
 /* -15 */     29154,     23254,     18705,     14949,     11916,
 /* -10 */      9548,      7620,      6100,      4904,      3906,
 /*  -5 */      3121,      2501,      1991,      1586,      1277,
 /*   0 */      1024,       820,       655,       526,       423,
 /*   5 */       335,       272,       215,       172,       137,
 /*  10 */       110,        87,        70,        56,        45,
 /*  15 */        36,        29,        23,        18,        15,
};
```

调度相关的系统调用

System Call	Description
`nice()`	Sets a process's nice value
`sched_setscheduler()`	Sets a process's scheduling policy
`sched_getscheduler()`	Gets a process's scheduling policy
`sched_setparam()`	Sets a process's real-time priority
`sched_getparam()`	Gets a process's real-time priority
`sched_get_priority_max()`	Gets the maximum real-time priority
`sched_get_priority_min()`	Gets the minimum real-time priority
`sched_rr_get_interval()`	Gets a process's timeslice value
`sched_setaffinity()`	Sets a process's processor affinity
`sched_getaffinity()`	Gets a process's processor affinity
`sched_yield()`	Temporarily yields the processor

代码例子

设置SCHED_FIFO和RT优先级

```
struct sched_param the_priority;

the_priority.sched_priority  = 50;
pthread_setschedparam(pthread_self(), SCHED_FIFO,
&the_priority);
```

工具chrt和renice

设置SCHED_FIFO和50 RT优先级

```
# chrt -f -a -p 50 10576
```

设置nice

```
# renice -n -5 -g 9394
# nice -n 5 ./a.out
```

课程练习源码

https://github.com/21cnbao/process-courses

更早课程

- 《Linux总线、设备、驱动模型》录播：
http://edu.csdn.net/course/detail/5329

- 深入探究Linux的设备树
http://edu.csdn.net/course/detail/5627

- Linux进程、线程和调度
http://edu.csdn.net/course/detail/5995

谢谢！