Completely Fair Scheduler (CFS) in Linux

November 20, 2019

Ellen Marie Andersen
Benjamin Cathelineau
Olexandr Matveyev

Our purpose:

Researching and understanding CFS

- What is it?
- What is the data structure?
- How does it work?
- Examples

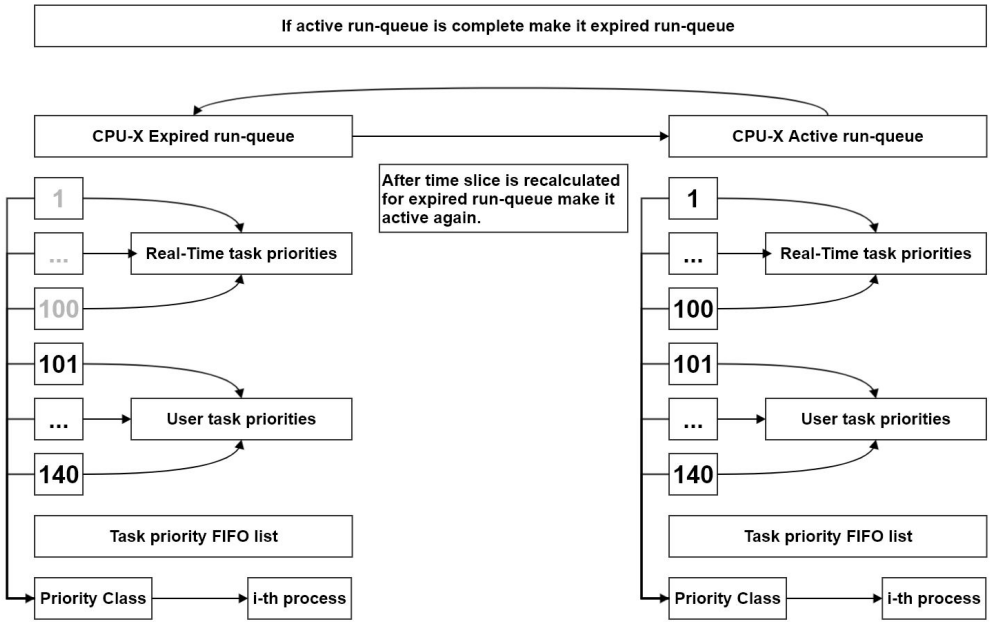
What we will cover today:

- Brief history of O(1)
- Why switching from O(1) to CFS?
- Fair scheduling
- CFS
- Niceness example

History & context

- O(1) replaced O(n)
- Incorporated into kernel starting from kernel version 2.6.0 till 2.6.22
- Replaced by CFS in 2007
- Why replaced?
  - Complex heuristics used to mark a task as interactive or non-interactive
- Interactive processes
  - Short tasks, Tasks that require frequent user interaction, Graphically intensive tasks
- Non-interactive
  - Any background process which is not required immediate input and will not produce any immediate output
- Author of O(1) and CFS is Ingo Molnar

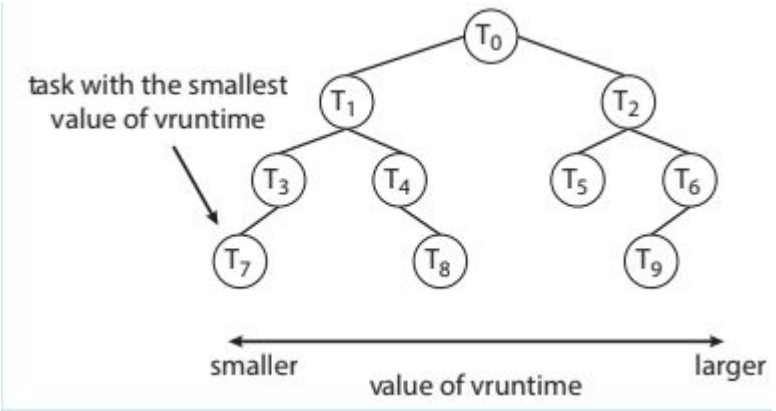
Brief explanation of how O(1) works



What is fair share scheduling?

- Sharing the processor in a “fair” way
- Example with 4 processes/groups, meaning N = 4 (all else equal)
- (100/N)% = 25% share per process/group
- Adjusted continuously
- How is this extended and implemented in CFS?

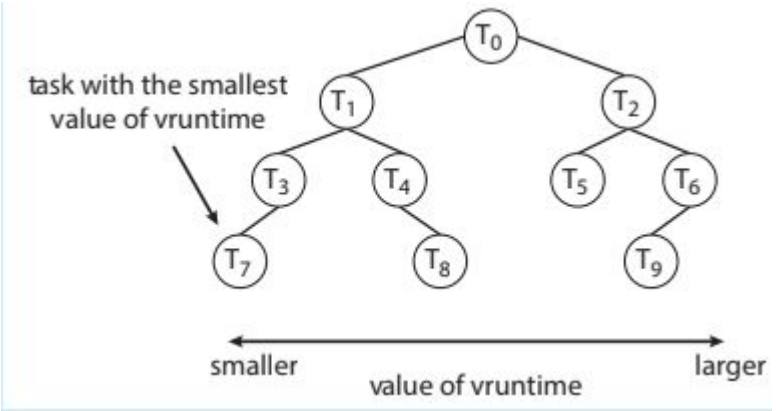
What is CFS?



- Efficient data structure (Red Black tree)
- Implemented in fair_sched_class
- vruntime decides where processes go in the tree
  - based on time spent executing (fair share!) and priority/niceness
  - I/O vs CPU bound processes
- What happens if one process spawns other processes to “take over” the CPU?

Source: Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating system concepts*. Hoboken, NJ: Wiley.

...Grouping scheduling!



- Members belong to same “task group”
- Session starts -> unique group membership id
- Fork() causes child to receive same group membership id as parent
- Each group treated as a task

```
0
/proc/sys/kernel/sched_autogroup_enabled (END)
```

```
1
/proc/sys/kernel/sched_autogroup_enabled (END)
```

Code function examples

- when a runnable task is to be placed in the red-black tree: enqueue_task(...)

- when a task is not runnable any longer, remove from tree: dequeue_task(...)

- when a new task is chosen to run(leftmost node): pick_next_task()

For those interested in reading more CFS code (10562 lines of code... ;)):

https://elixir.bootlin.com/linux/latest/source/kernel/sched/fair.c

What is CFS continued

- Virtual runtime:
- Priorities
- Niceness

Very nice example

“In the current implementation, each unit of difference in the nice values of two processes results in a factor of **1.25** in the degree to which the scheduler favors the higher priority process.”

The man page.

benjamin@debian-benjamin-laptop: ~

File Edit View Search Terminal Help

```
benjamin@debian-benjamin-laptop:~$ nice -n 19 ping debian.org -i 10
PING debian.org (128.31.0.62) 56(84) bytes of data.
64 bytes from mirror-csail.debian.org (128.31.0.62): icmp_seq=1 ttl=50 time=156 ms
```

```
  1  : 33.8% sys:  6.0% low:  4.0%          Tasks: 152, 778 thr; 1 running
  2  : 39.9% sys:  2.6% low:  1.3%          Load average: 1.68 1.29 1.19
  3  : 46.5% sys:  2.6% low:  5.2%          Upt[|                 2 days, 13:35:19]
  4  : 43.5% sys:  3.2% low:  2.6%
  Mem:7.69G used:6.19G buffers:78.0M cache:876M
  Swp:7.46G used:132M

   PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
21392 benjamin   39  19 11240  2936  2720 S  0.0  0.0  0:00.00 ping debian.org -i 10

EnterDone  EscClear   Filter: ping
```

```
benjamin@debian-benjamin-laptop:~$ ping centos.org -i 10
PING centos.org (85.12.30.226) 56(84) bytes of data.
```

```
benjamin@debian-benjamin-laptop:~$
```

```
[0] 0:bash*                                                    "debian-benjamin-lapto" 10:08 18-Nov-19
```

Beyond niceness : chrt

```
benjamin@debian-benjamin-laptop:~$ chrt --max
SCHED_OTHER min/max priority    : 0/0
SCHED_FIFO min/max priority     : 1/99
SCHED_RR min/max priority       : 1/99
SCHED_BATCH min/max priority    : 0/0
SCHED_IDLE min/max priority     : 0/0
SCHED_DEADLINE min/max priority : 0/0
```

-policy change :

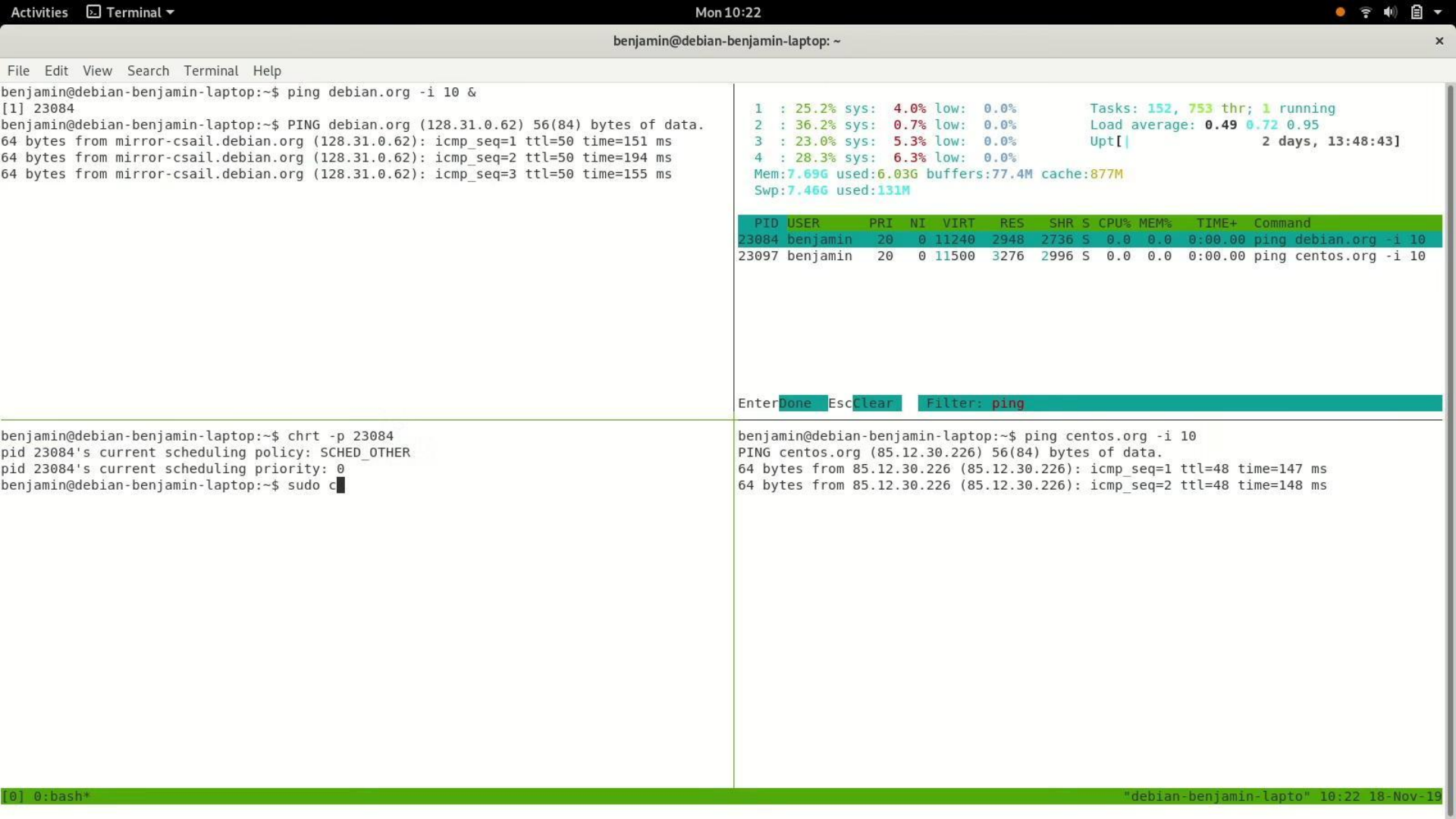
Real time processes, don’t uses CFS, uses priority :

SCHED_RR, SCHED_FIFO, SCHED_DEADLINE

Normal processes, use niceness: to calculate priority

SCHED_OTHER, SCHED_IDLE, SCHED_BATCH

-priority change


Activities
Terminal
Mon 10:22
benjamin@debian-benjamin-laptop: ~
File Edit View Search Terminal Help
benjamin@debian-benjamin-laptop:~$ ping debian.org -i 10 &
[1] 23084
benjamin@debian-benjamin-laptop:~$ PING debian.org (128.31.0.62) 56(84) bytes of data.
64 bytes from mirror-csail.debian.org (128.31.0.62): icmp_seq=1 ttl=50 time=151 ms
64 bytes from mirror-csail.debian.org (128.31.0.62): icmp_seq=2 ttl=50 time=194 ms
64 bytes from mirror-csail.debian.org (128.31.0.62): icmp_seq=3 ttl=50 time=155 ms
1 : 25.2% sys: 4.0% low: 0.0%
2 : 36.2% sys: 0.7% low: 0.0%
3 : 23.0% sys: 5.3% low: 0.0%
4 : 28.3% sys: 6.3% low: 0.0%
Tasks: 152, 753 thr; 1 running
Load average: 0.49 0.72 0.95
Upt[2 days, 13:48:43]
Mem:7.69G used:6.03G buffers:77.4M cache:877M
Swp:7.46G used:131M
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
23084 benjamin 20 0 11240 2948 2736 S 0.0 0.0 0:00.00 ping debian.org -i 10
23097 benjamin 20 0 11500 3276 2996 S 0.0 0.0 0:00.00 ping centos.org -i 10
EnterDone EscClear Filter: ping
benjamin@debian-benjamin-laptop:~$ chrt -p 23084
pid 23084's current scheduling policy: SCHED_OTHER
pid 23084's current scheduling priority: 0
benjamin@debian-benjamin-laptop:~$ sudo c
benjamin@debian-benjamin-laptop:~$ ping centos.org -i 10
PING centos.org (85.12.30.226) 56(84) bytes of data.
64 bytes from 85.12.30.226 (85.12.30.226): icmp_seq=1 ttl=48 time=147 ms
64 bytes from 85.12.30.226 (85.12.30.226): icmp_seq=2 ttl=48 time=148 ms
[0] 0:bash*
"debian-benjamin-lapto" 10:22 18-Nov-19

Shell script test : What are the policies of the processes running on the system

```
#! /bin/bash
ps -aux |grep [0-9]|awk '{print $2}'  > test.txt
cat test.txt |while read line
do
chrt  -p $line 2> /dev/null
done
```

```
CSCI460/project$ ./shellscript |grep IDLE|wc
      1       6      49
```

```
benjamin@debian-benjamin-laptop:~/Documents/montana
CSCI460/project$ ./shellscript |grep OTHER |wc
    218    1308   10833
```

```
benjamin@debian-benjamin-laptop:~/Documents/monta
CSCI460/project$ ./shellscript |grep FIFO |wc
     12      72     572
```

```
le@elle-Inspiron-7348:~/Downloads$ ./shellscript | grep OTHER | wc
 245    1470   12154
```

```
lle@elle-Inspiron-7348:~/Downloads$ ./shellscript | grep FIFO | wc
   15      90     713
```

Example of a SCHED_FIFO process

```
benjamin@debian-benjamin-laptop:~/Documents/montana2019/1nd semester/courses documents/CSCI460/project$ ps aux | grep migration
root         12  0.0  0.0      0     0 ?        S    Nov15   0:01 [migration/0]
root         16  0.0  0.0      0     0 ?        S    Nov15   0:01 [migration/1]
root         21  0.0  0.0      0     0 ?        S    Nov15   0:00 [migration/2]
root         26  0.0  0.0      0     0 ?        S    Nov15   0:00 [migration/3]
benjamin 24402  0.0  0.0   6208   884 pts/7    S+   12:15   0:00 grep migration
benjamin@debian-benjamin-laptop:~/Documents/montana2019/1nd semester/courses documents/benjamin@debian-benjamin-laptop:~/Documents/
benjamin@debian-benjamin-laptop:~/Documents/montana2019/1nd semester/courses documents/CSCI460/project$ chrt -p 12
pid 12's current scheduling policy: SCHED_FIFO
pid 12's current scheduling priority: 99
```

What we learned

- History
- Fair share scheduling
- CFS: what is it and how it works
- vruntime: priority,niceness and number of processes
- change real time(chrt), scheduling policies

Thank you!