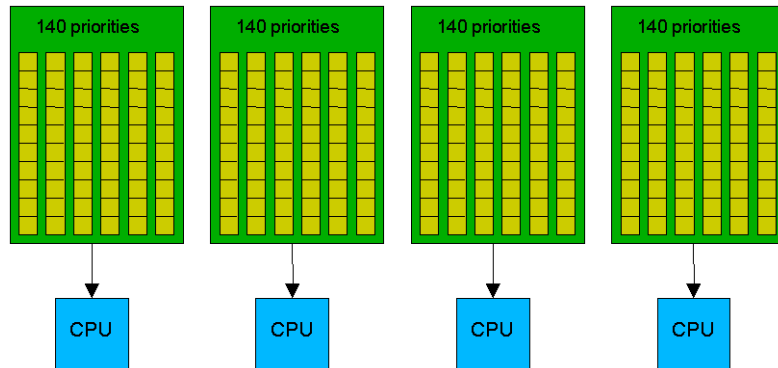




## Linux O(1) Scheduler (7)

for each CPU and each priority one queue (that is: 140 lists per CPU)!



Picture: Linux Journal,  
<http://www.linuxjournal.com/node/7178/>

## Linux O(1) Scheduler (9)

in addition to the Runqueue there is an extra „Expired Runqueue“

- active process whose time quantum runs out will be preempted and moved to the Expired Queue
- while moving, the scheduler recalculates quantum and priority for this process (i.e. possibly sorts it into a different priority level).
- when the Runqueue is completely emptied, swap Runqueue and Expired Runqueue

## Linux O(1) Scheduler (8)

Finding the next process is very easy:

- each CPU only has to search its private process list
- Bitmap stores information which (of the 140) queues are empty – a search of the kind „1st bitmap field with value 1“ is quick
- within the found list pick the first process
- search time depends „on 140“, but not on the number of processes -> O(1)

## Linux O(1) Scheduler (10)

**Interactivity estimator**

- scheduler tries to find out whether processes are I/O bound or CPU bound
  - metric: proportion of compute time and (I/O) waiting time
  - scheduler
    - awards I/O bound processes
    - punishes CPU bound processes
- up to +/- 5 points for priority calculation

# Linux O(1) Scheduler (11)

## Load Balancer

- actually: try to avoid CPU migration since CPU cache becomes unusable
- on the other hand: CPUs with long idle times are even worse
- every 200 ms some CPU checks whether the load distribution is unbalanced; if so, processes are redistributed
- problem: treatment of HyperThreading CPUs with virtual CPUs

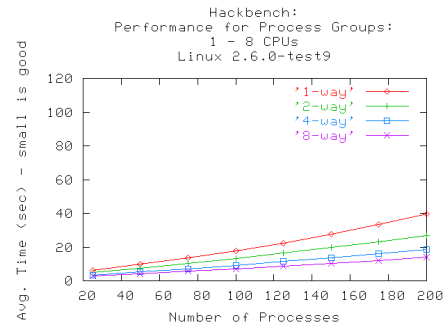
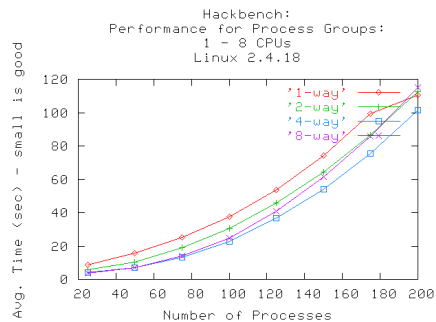
```

Sep 19 14:20:18 amd64 sshd[20494]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61557
Sep 19 14:27:43 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 01:00:01 amd64 /usr/sbin/cron[29278]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 20 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 02:00:01 amd64 /usr/sbin/cron[31031]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 20 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:46:44 amd64 sshd[6516]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62004
Sep 20 12:46:44 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:48:41 amd64 sshd[6609]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62105
Sep 20 12:54:44 amd64 sshd[6694]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62514
Sep 20 15:27:35 amd64 sshd[9077]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64242
Sep 20 15:27:35 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:37:11 amd64 sshd[10102]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63375
Sep 20 16:37:11 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:38:10 amd64 sshd[10140]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63546
Sep 20 01:00:01 amd64 /usr/sbin/cron[17055]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 20 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 02:00:01 amd64 /usr/sbin/cron[17978]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 21 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:43:26 amd64 sshd[31088]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63397
Sep 21 17:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:53:39 amd64 sshd[31269]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64391
Sep 21 18:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 19:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 01:00:01 amd64 /usr/sbin/cron[4674]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 22 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 02:00:01 amd64 /usr/sbin/cron[5493]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 22 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 20:23:21 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 01:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 23 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 02:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 23 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:05 amd64 sshd[6541]: Accepted publickey for esser from ::ffff:192.168.1.5 port 59771 ssh2
Sep 23 18:04:05 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:34 amd64 sshd[6606]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62093
Sep 24 01:00:01 amd64 /usr/sbin/cron[12436]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 24 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 02:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 24 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 11:15:48 amd64 sshd[20998]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64456
Sep 24 11:15:48 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 13:49:08 amd64 sshd[21397]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61330
Sep 24 13:49:08 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_midi_event: unsupported module, tainting kernel.
Sep 24 15:42:07 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_oss: unsupported module, tainting kernel.
Sep 24 20:25:31 amd64 sshd[29399]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62566
Sep 24 20:25:31 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 01:00:02 amd64 /usr/sbin/cron[6621]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 25 01:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 02:00:01 amd64 /usr/sbin/cron[1484]: (root) CMD (/sbin/evlogmgr -c 'age > *30d*')
Sep 25 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:25 amd64 sshd[8889]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64183
Sep 25 10:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:47 amd64 sshd[8921]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64253
Sep 25 11:30:02 amd64 sshd[9372]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62029
Sep 25 11:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:05:37 amd64 sshd[11553]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62822
Sep 25 14:05:37 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:06:10 amd64 sshd[11586]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62951
Sep 25 14:07:13 amd64 sshd[11608]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63392
Sep 25 14:08:33 amd64 sshd[11620]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63709
Sep 25 15:25:33 amd64 sshd[12930]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62778
    
```

# Desktop Scheduling

# Performance Linux 2.4 / 2.6

Hackbench: up to 200 client / server processes



Picture: <http://developer.osdl.org/craiger/hackbench/>

# „Desktop Scheduling“

## Multimedia Scheduler

- most schedulers differentiate classically:
  - I/O bound (interactive) vs.
  - CPU bound (non-interactive)
- low CPU utilization -> high priority
- problem: multimedia applications (video, games) require a lot of CPU time, i.e., they are indistinguishable from classical background jobs -> they will not perform well

# „Desktop Scheduling“

## Performance Test:

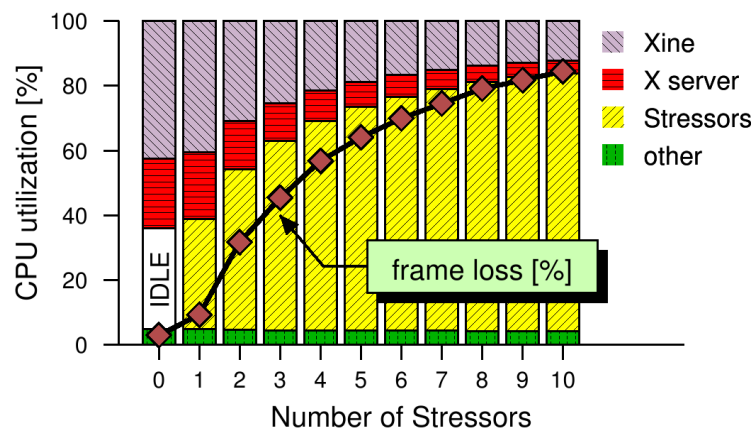
- „Stressor applications“: very CPU bound, e.g. Kernel compilation
- how do multimedia applications behave when the number of stressor processes grows?

# „Desktop Scheduling“

## Old approaches for recognizing interactivity:

- programs make statements about themselves („I am interactive/not interactive“)
- user decides (via start parameters or with a process monitor) which programs should run with a high priority
- program window with focus (active) receives more compute time (MS Windows)

# „Desktop Scheduling“



classical scheduling (picture: [1])

[1] Elision, Tsafir, Feitelson, „Desktop Scheduling: How can we know what the user wants?“, 14th ACM Intl. Workshop on Network & Operating Systems Support for Digital Audio & Video (NOSSDAV), pp. 110-115, Jun 2004, www.csie.nyu.ac.il/~dantis/papers/HUCpr04NOSSDAV.pdf

# „Desktop Scheduling“

## New approach for recognizing interactivity:

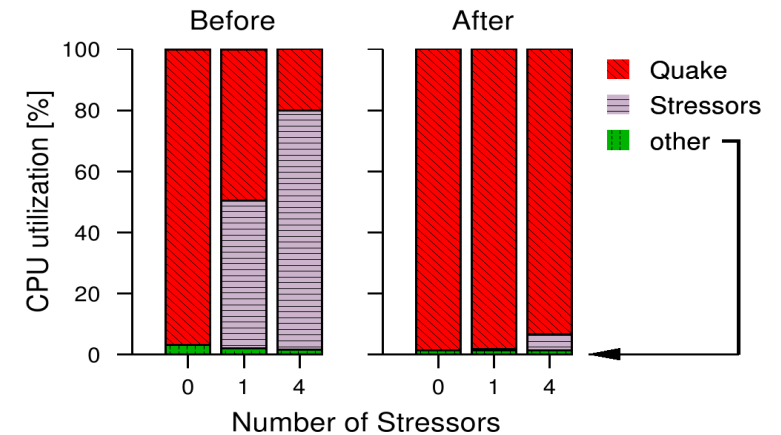
- some I/O devices are very recently used by interactive programs, e.g.
  - keyboard: obviously interactive
  - video card: when a process permanently updates a large screen area, then that is also assumed to be interactive
- statistical data of the I/O devices tell the scheduler which processes are „relevant“ for the user

# „Desktop Scheduling“

## HuC (human centered) Devices

- only selected I/O devices are interesting, e.g. keyboard, mouse, display, joystick, sound card -> **HuC Devices**
- for the screen: patch the X server
  - collect client-specific I/O traffic data
  - forward them 1x/sec to the scheduler
  - measure: what fraction of the screen size has been changed?

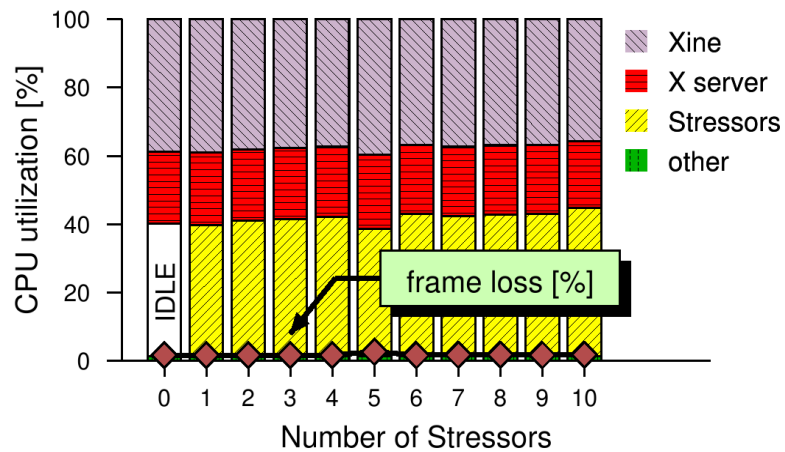
# „Desktop Scheduling“



how much compute time does Quake receive?  
left: normal Linux scheduler, right: HuC scheduler

Bild: [1]

# „Desktop Scheduling“



HuC scheduler leads to better results

Picture: [1]