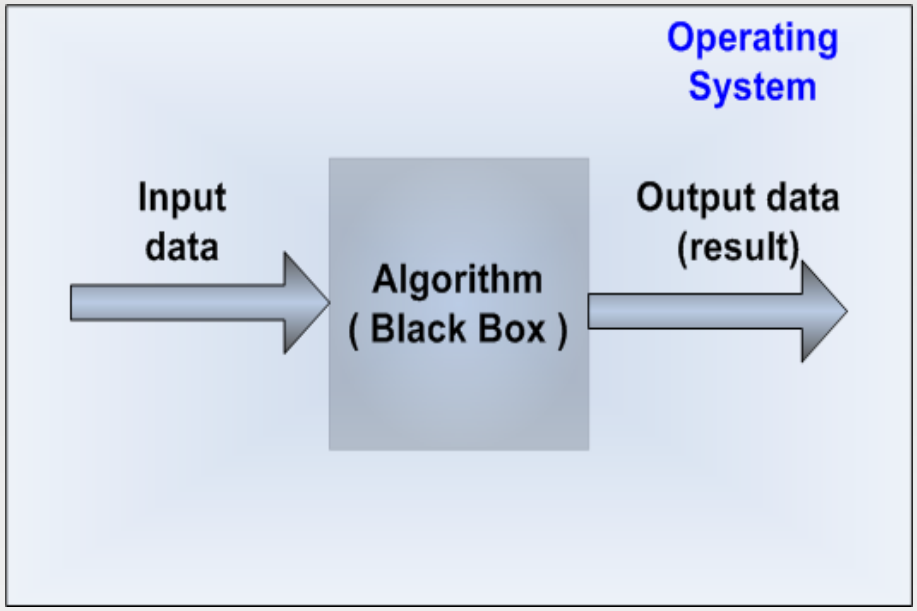
Optimizing the performance of the algorithms block data processing in Linux

Ovseenko Anton, SUAI
Victor Minchenkov, SUAI
Alexander Povalyaev, EMC

27 april 2011

Problem statement

1. It is known that the program does, but there is no source code.
2. Required to optimize the running time by changing the settings of operating system and input data.



3. Other system processes should not degrade.
4. Consider only modern architecture and kernel version (kernel 2.6, multiprocessor hardware systems and 64-bit operating system).

What can optimize?

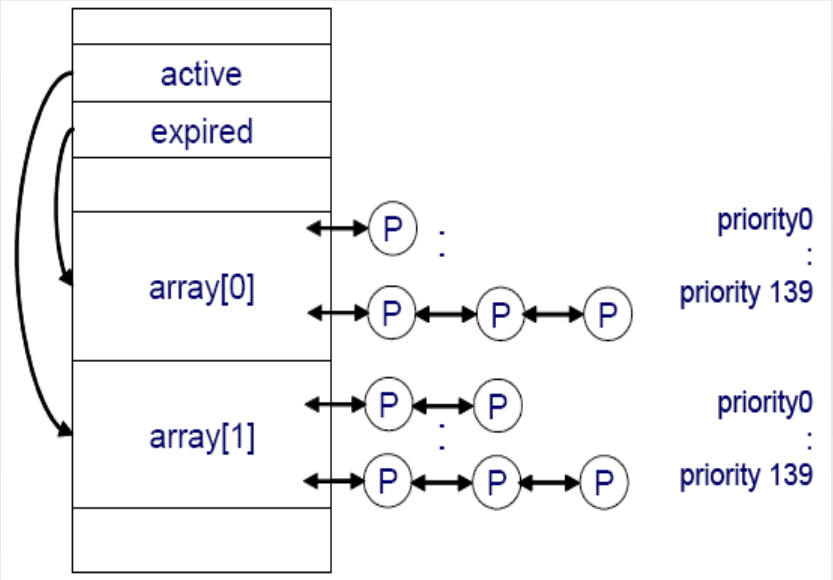
Methods to accelerate execution of the program

- Task scheduler (priority)
- Input/Output scheduler (priority)
- Use huge memory pages
- Work at the kernel space (Driver OS)

Task scheduler

O(1) – Appeared in 1993.

2 queues processes (sleeping and active), 140 priorities. Now contains **7000!** lines of C code.



CFS - Completely Fair Scheduler (kernel 2.6.23 from 9.10.2007).

You can not control the priority of the process directly, we can only influence it:

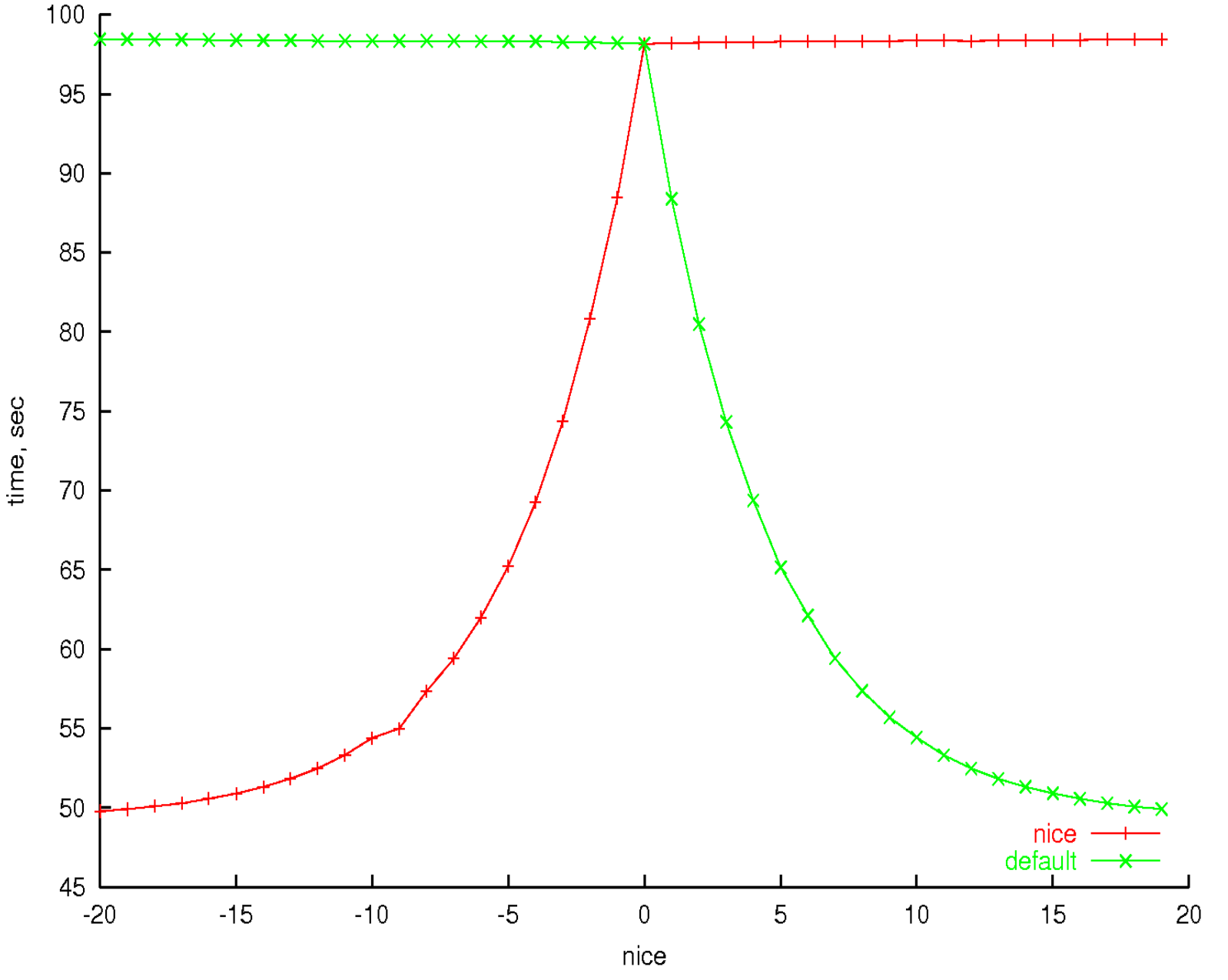
```
nice nice_value program

renice nice_value PID
```

Nice value: [-20, 19]

* ”Understanding the Linux Kernel” byDaniel P. Bovet, Marco Cesati

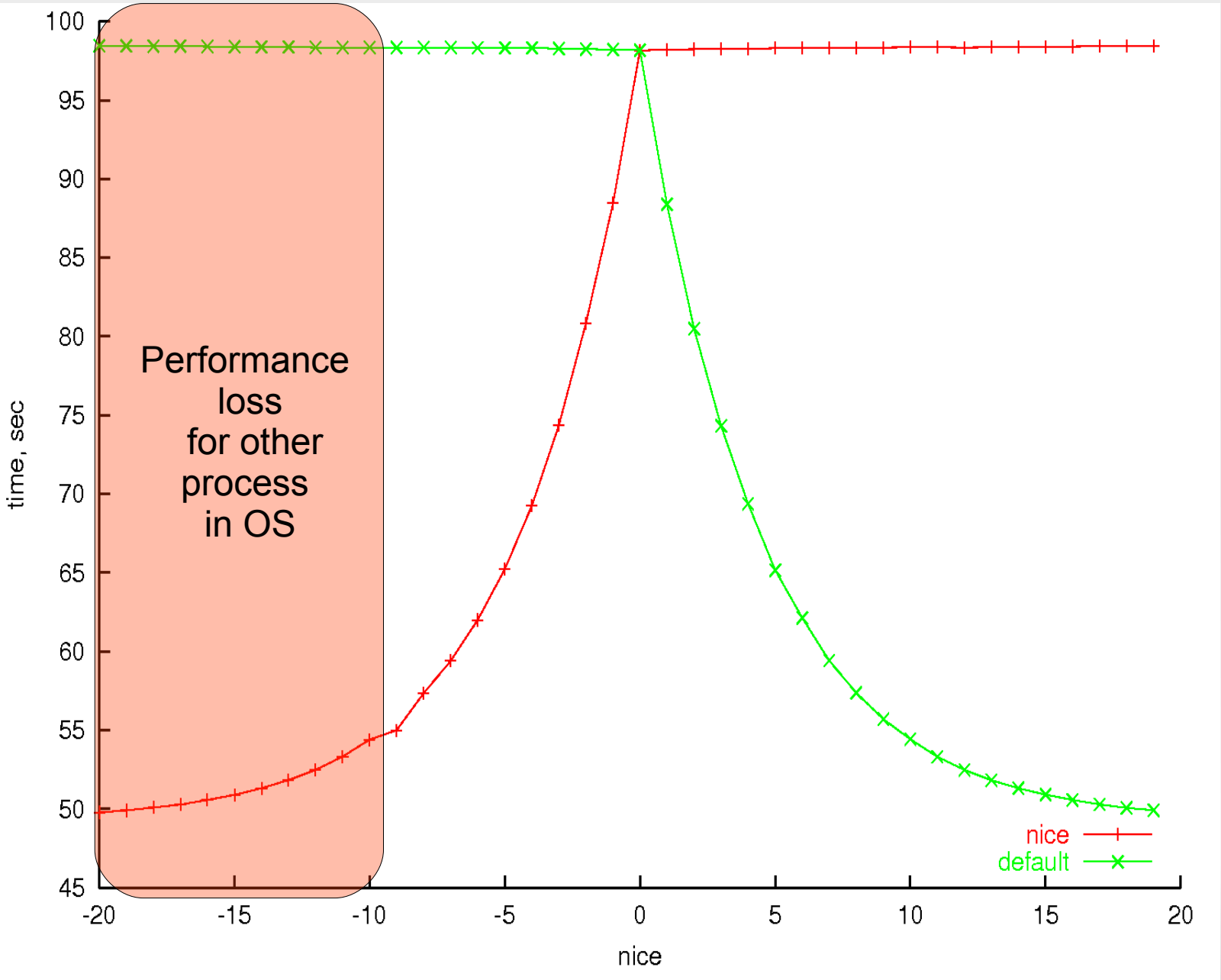
Task scheduler
(calculation of pi, 60000 characters, 2 processes)



nice (red line) -
nice in x-axis
(-20, 19)

default (green line)
nice = const(0)

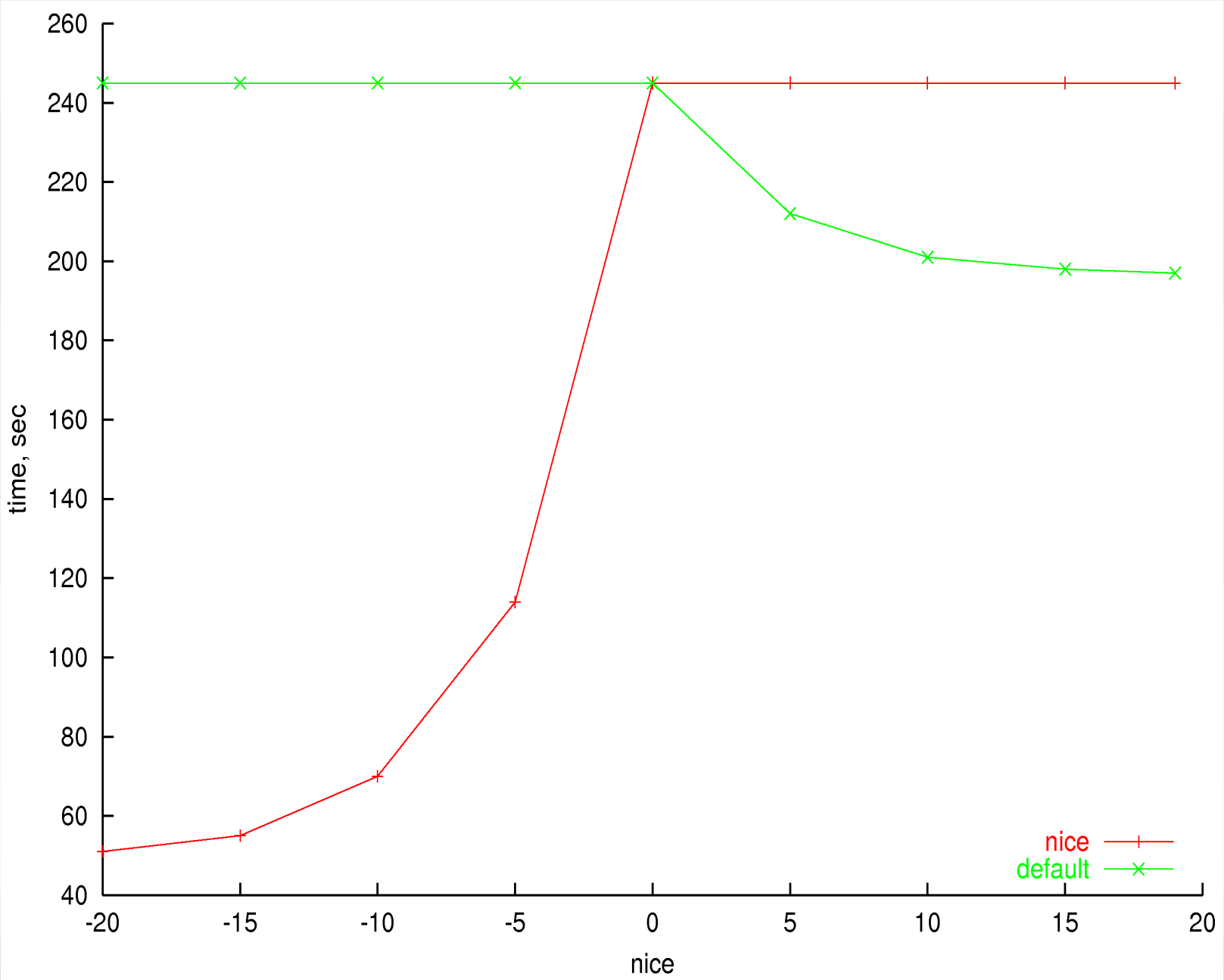
Task scheduler
(calculation of pi, 60000 characters, 2 processes)



nice (red line) - nice in x-axis (-20, 19)

default (green line) nice = const(0)

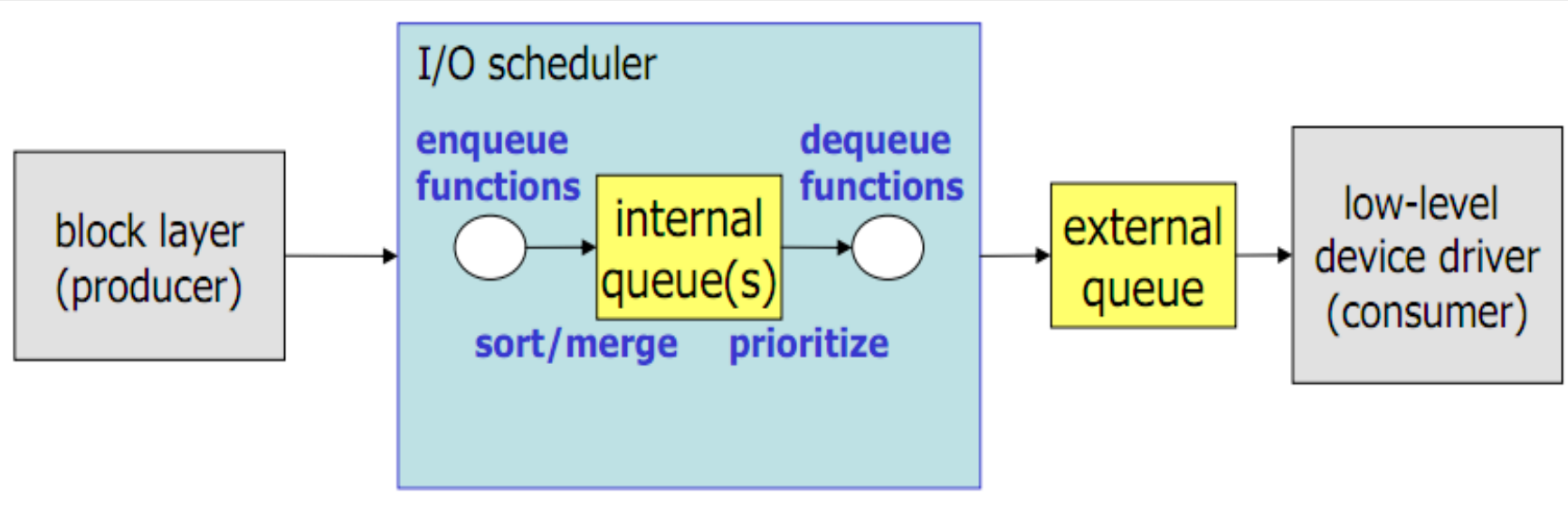
Task scheduler
(calculation of pi, 60000 characters, 5 processes)



nice (red line) - nice in x-axis (-20, 19)

default (green line) Groups of 4 Processes, nice = const(0)

Input/Output scheduler



*

Input/Output schedulers operates:

- Merger - process of adopting two or more adjacent Input/Output requests, and combining them into one request.
- Sorting - the process of ordering Input/Output requests.

* Linux I/O schedulers, Hao-Ran Liu

Input/Output scheduler

I/O schedulers installed in the kernel (Fedora 14)

- Complete Fair Queueing (CFQ) – default (c 2.6.18)
- NO-OP [USB, SSD].

  Useful when there is no cost to the mechanical movement.
- Deadline

  [To speed up the reading from the disk uses the principle of lazy writing].

  Is useful for distributed queries to the disk (database).
- Anticipatopy - not in the current kernel

  [Tries to "guess" next user action]. Minimizes disk head movement.

Input/Output scheduler

CFQ

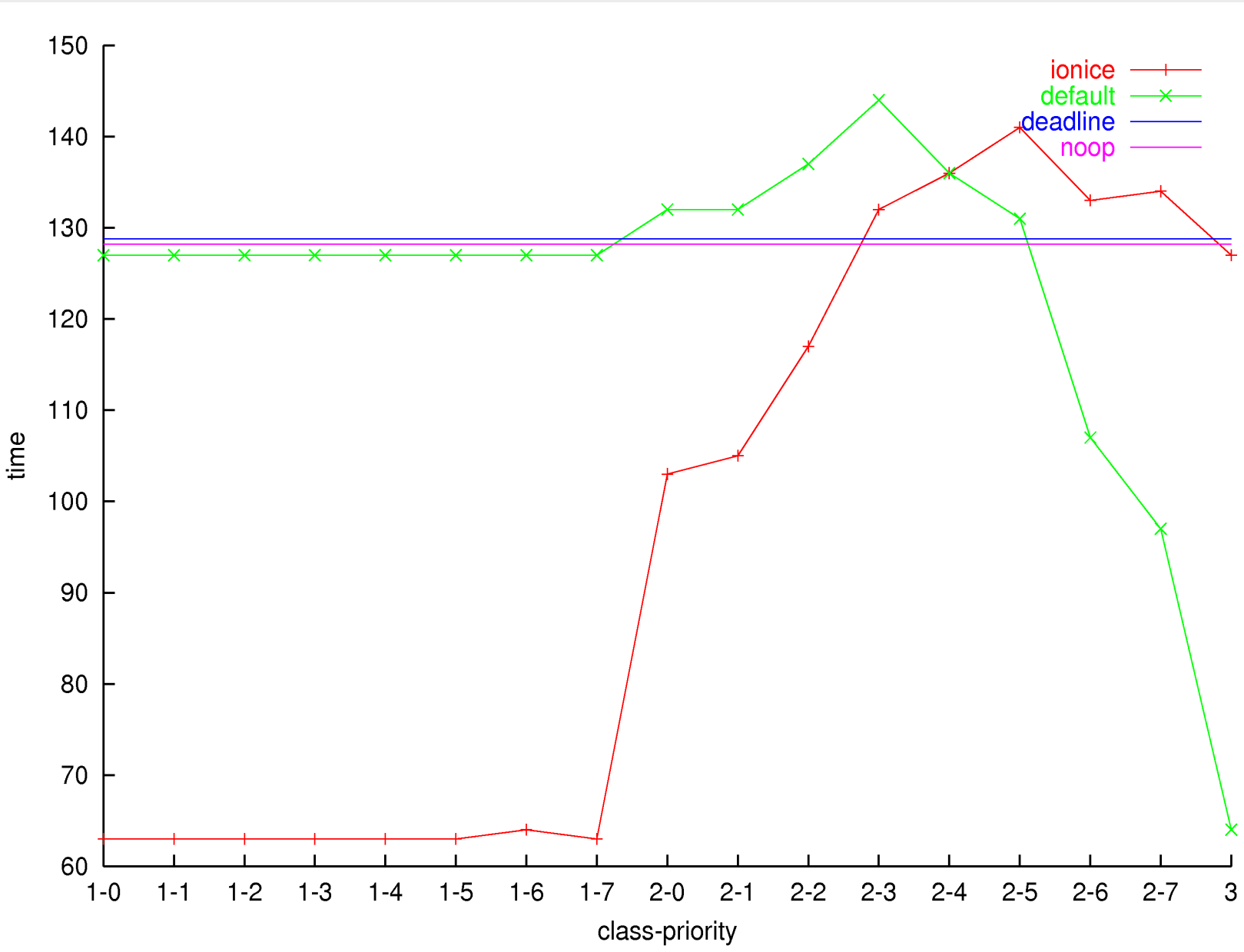
possible management priority process

```
ionice -c class -n priority -p program
iorenice -c class -n priority -p PID
```

Priority class

- 1 - Real time – [0-7]
- 2 - Best Effort – [0-7] - default (2 class, priority 4)
- 3 - Idle

Input/Output scheduler
(character by character reading a file size of 2 Gb)



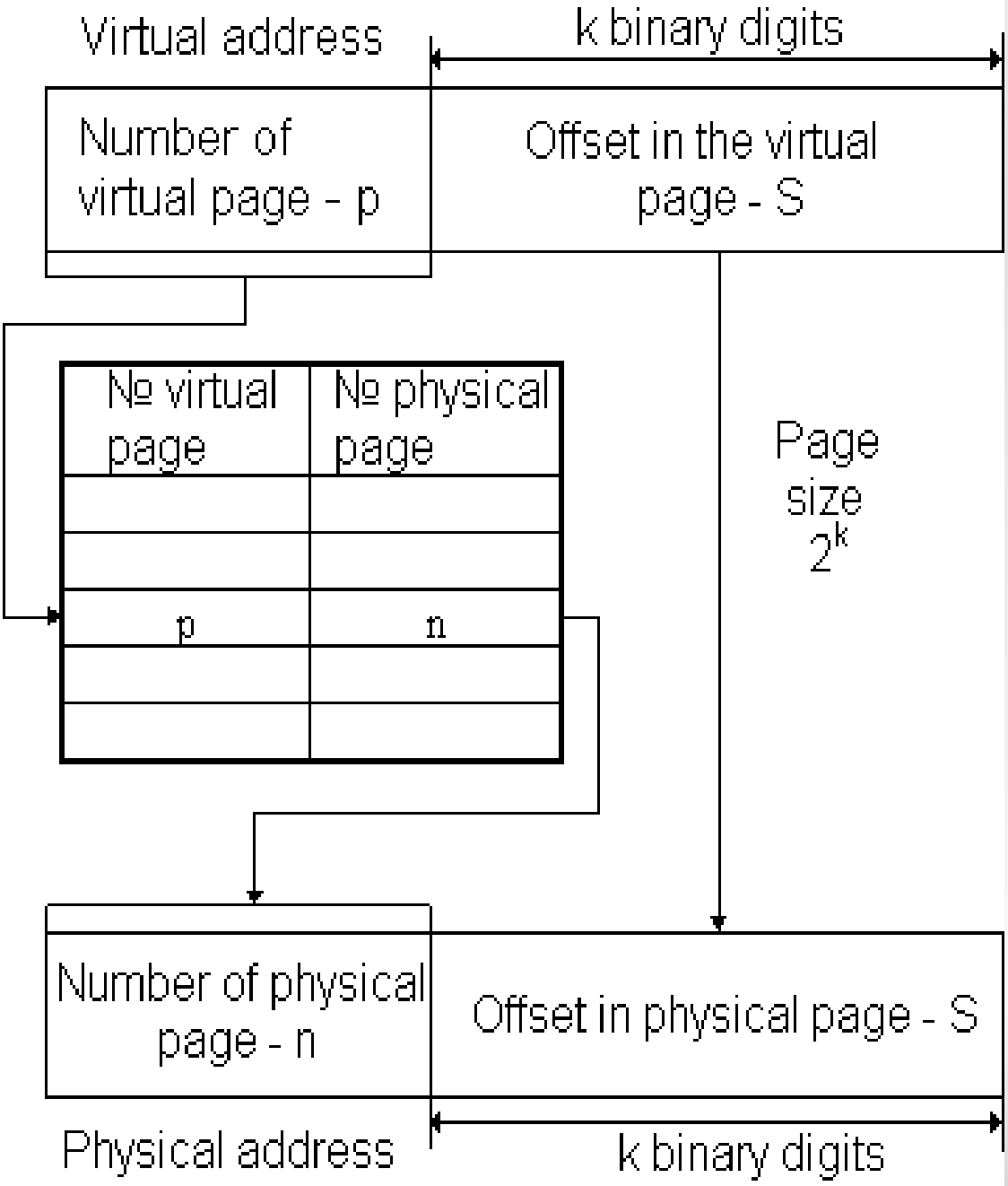
ionice
(CFQ,
nice in x-axis)
(-20, 19)

default
(CFQ, nice = 0)

deadline
(2 processes,
in 1 line)

noop
(2 processes,
in 1 line)

Memory paging



Linux System memory is organized as pages of the volume of 4K.

If the memory is completely depleted, the OS will look for a long time unused memory pages to move them from memory to disk. If any of these pages is required, Linux restore them from disk.

Huge pages

4 Kb – the size of standard memory pages.

2 Mb (x86) or 4 Mb (x86_64) – the size of huge page memory.

1) array 4 Mb = 4 Kb * 1024 pages

or 4 Mb = 2 Mb * 2 pages

2) TLB (Translation Lookaside Buffer) – buffer that caches the last several transformations of memory addresses.

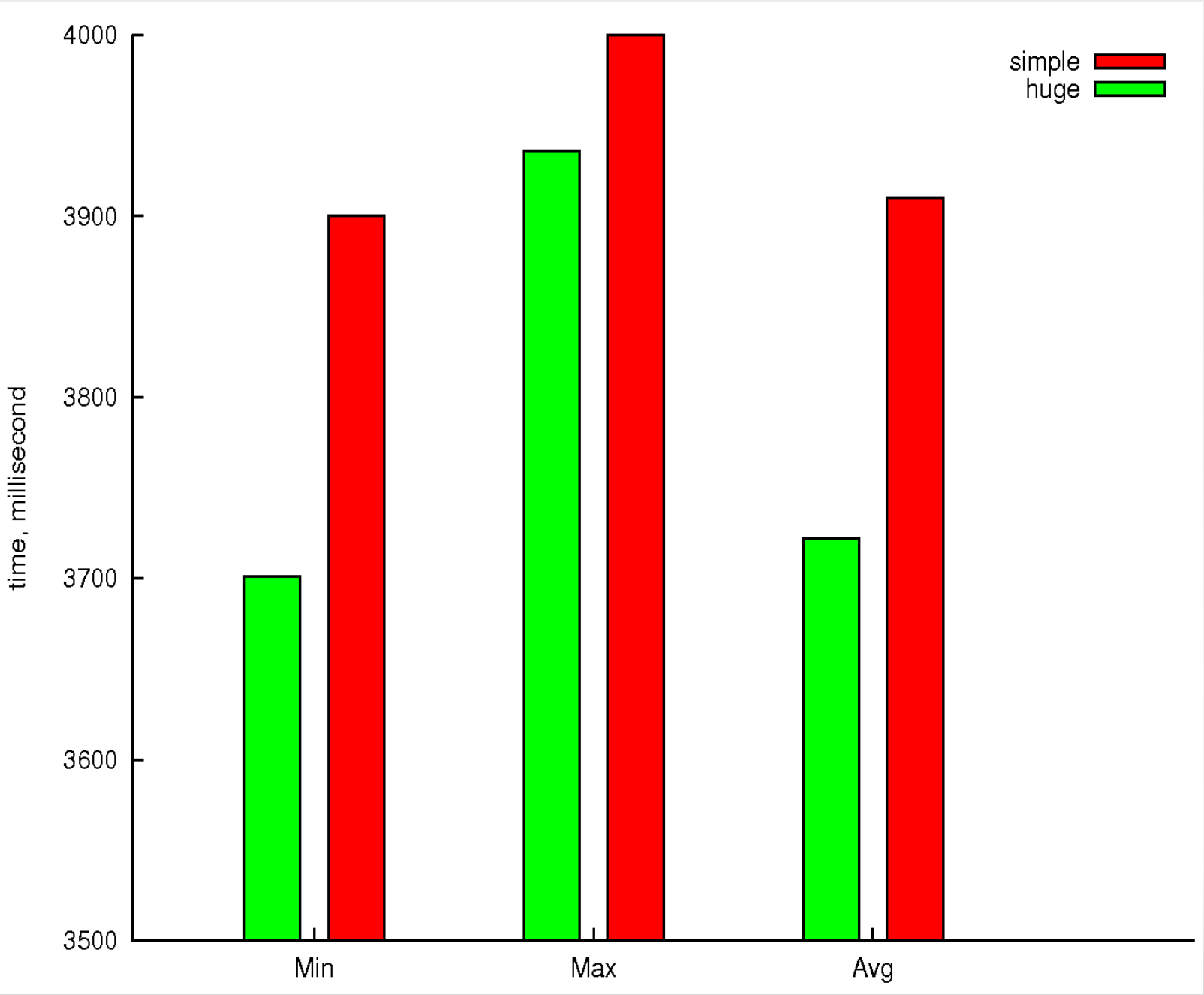
The advantages of using huge pages:

+ fewer pages required to allocate.

+ search in the TLB is faster (as TLB caches the few conversions)

Huge pages. Results

(Copy memory from the input of the output, 500 Mb)

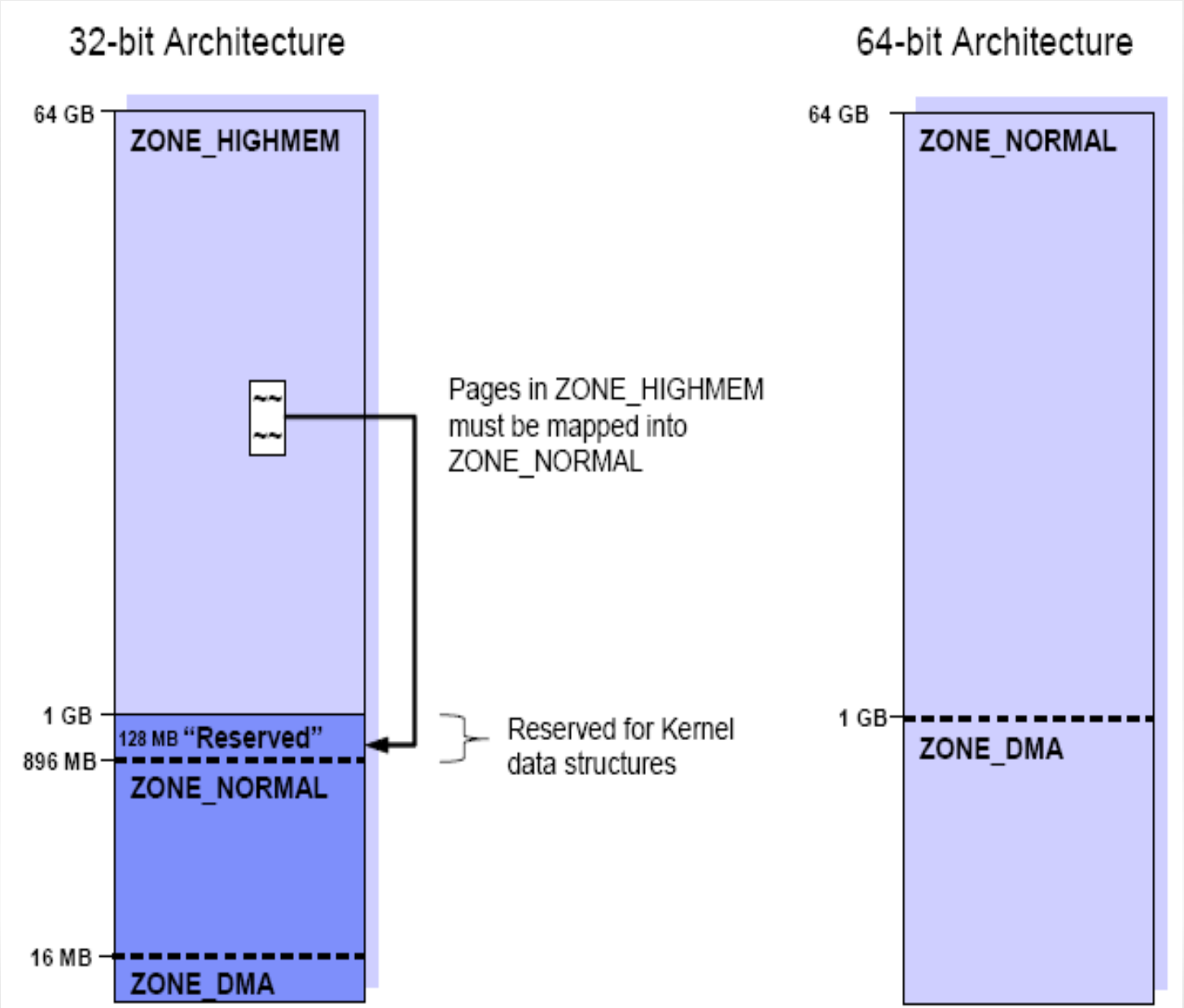


Gain on huge pages:

5 %

Work at the kernel space

Linux. Memory Architecture



*

* "Linux Performance and Tuning Guidelines", IBM Redpaper

Conclusions

- **Getting the win without interfering with the code of the programs is not possible (increasing the priority of our process to the scheduler reduces the interactivity of the other processes).**
- **On the example of huge pages shows that the gain of 5% can take place.**
- **Kernel space and DMA: future work.**

Thank you for your attention!

Questions?