No more latencies!

Paolo Valente, Linus Walleij, Ulf Hansson

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Agenda

- A tale of lag and high latency
- A cure: the BFQ I/O scheduler
- How does BFQ make it?
- BFQ is landing into blk-mq: BFQ-MQ
- What about Android?
- A look into the future
- Support for blk-mq in MMC
- CMDQ support for eMMC

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

A tale of lag and high latency

- Storage devices are fast and will become faster and faster
- Latencies due to I/O are finally negligible!
- What about reddit to check people enthusiasm about that?
  - Um, hot topic in last August: Why is Android Lag Buildup Problem?
- Ok, let's check concretely
- Let's just start a common app, Facebook, under Android on a HiKey
  - First without any other I/O in progress
  - Then with some I/O in background, mimicking, e.g., some other app being updated
  - Demo made by Luca Miccio

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Houston, we have a problem

- So, lag is extremely high!
- Is this only an Android problem?
- No, it affects PC Linux distributions too
- We are about to see it with a demo for HDDs
  - We will see both the problem and a solution at work: the BFQ I/O scheduler
- No time to see it for SSDs too, but
  - You can find, e.g., here, a demo showing that the problem affects those fast devices as well
  - As a final remark, we will quickly highlight why the problem doesn't get solved in spite of a much higher speed

Linaro
connect
Budapest 2017

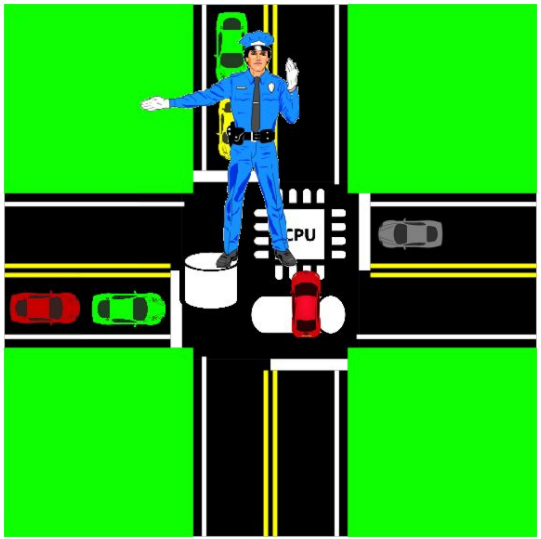
ENGINEERS
AND DEVICES
WORKING
TOGETHER

A cure for latency, and not only: BFQ

- BFQ is an I/O scheduler
- Before digging into details, probably better to provide some minimal background
  - What are I/O schedulers?
  - Where are they?
    - blk: the legacy block layer
    - blk-mq: the new, multi-queue block layer

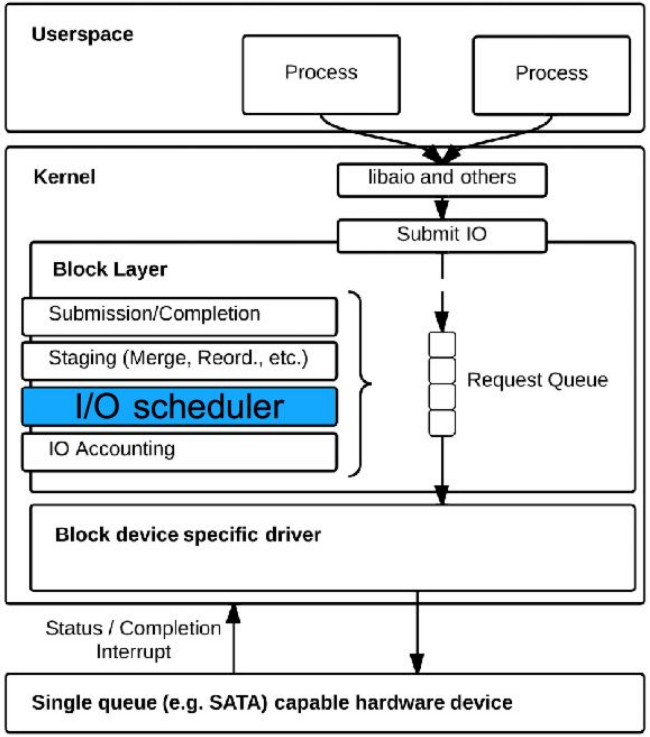
I/O schedulers

- Block-layer components, deciding the order in which I/O requests are to be served
  - Especially if several processes compete for the same storage device
- Order chosen so as to guarantee:
  - High I/O throughput
  - Low latency
  - High responsiveness,
  - Fairness



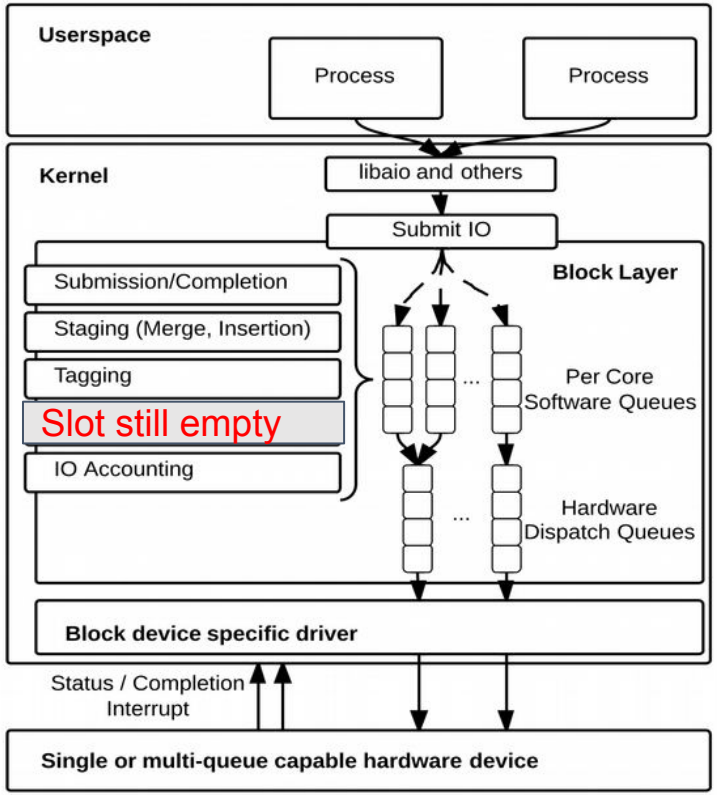
blk and I/O schedulers

- Three I/O schedulers available in blk
  - CFQ, DEADLINE and NOOP
- They decide the order in which I/O requests are Inserted into the request queue
- Note that there is only one request queue
- This makes it impossible to keep the pace with fast, highly parallel storage devices



blk-mq and (no) I/O schedulers

- blk-mq overcomes the limitations of blk by turning the single request queue of blk into multiple, independent queues
- Yet, no I/O scheduler available yet



Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Turning back to BFQ

- Motivation for BFQ
  - As we are going to see, blk schedulers are not very successful
- BFQ **really** guarantees
  - High responsiveness
  - Low latency to soft real-time applications
    - No more frame drops!
  - High throughput
- Ok, time to see BFQ at work
- Let’s go on with the Android demo (blk) …
- And the PC demo (still blk)
  - Youtube address
- Then some details on how BFQ makes it
- Finally a short history of the project

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

How does BFQ work?

- BFQ can be divided into two main components
  - Accurate proportional-share (weight-based) scheduling engine
    - Guaranteeing to each process its share of the throughput
  - Set of low-latency heuristics
    - Assign a high weight to applications to privilege
- How does BFQ make it to guarantee such a low latency?
  - By using the throughput for the right I/O requests
- If needed for latency
  - BFQ sacrifices throughput
  - But not that much: as opportunistic as possible
  - The actual added value is guaranteeing both a low latency and a high throughput at the same time
- When there is no critical applications to privilege
  - BFQ aims only at maximizing throughput

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

BFQ homepage

- Full details on BFQ can be found in its homepage
  - More details on how it works
  - Usage instructions
  - Pointers to demos
  - Pointers to existing papers on BFQ
  - Results of extensive tests (graph) on
    - HDDs (both single and RAID), SSDs, eMMCs, SD-CARDs, micro HDDs, ...
  - Pointer to the benchmark suite devised and used for the tests
  - Patches for kernel versions up to 4.10 (see next slides)
  - Pointers to package archives for various distributions

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Short history 1/2

- BFQ originates from a scientific paper by Fabio Checconi and Paolo Valente, back in 2008
- In the same year Fabio implemented BFQ's very first version
- Many enhancements made after then, mainly by Paolo, but also by other contributors (Arianna Avanzini, Mauro Andreolini, ...)
- Currently available as a legacy I/O scheduler non-upstream
  - Adopted in various distributions (ArchLinux ARM, Sabayon, Manjaro, ….)

Short history 2/2

- Submitted for upstreaming in blk several times over the last 6 years
  - Always rejected, for reasons unrelated with performance and quality
- Meanwhile, blk-mq has taken over, and blk has become legacy
  - blk could not be touched any more
  - Yet, relevant developers asked to have BFQ directly in blk-mq!
  - Jens Axboe offered to provide the needed framework

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Current status: BFQ-MQ

- A collaboration with BLOCK maintainers and developers is in progress
- Jens' framework is basically complete
- A testing version of BFQ-MQ is available
  - Based against 4.10
- Aiming for v4.12!
- Come see a first demo of BFQ-MQ on Friday!
- We will also show BFQ at work on the HiKey
- Both for BFQ and BFQ-MQ demos, we will show
  - Throughput
  - Responsiveness
  - Latency for soft real-time (frame drop in video playing)

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

What about Android?

- Linaro is supporting mainly the BFQ-MQ project
  - Far from reach for Android
- Results with Android are very exciting
  - Lag is evidently a serious problem
- Work to do is not trivial
  - First step: backport of last BFQ version to 4.4 and/or 4.9
  - Not sufficient, for the higher relevance of VM-related issues, further important steps are needed
- Trying to push for support with
  - Google Android Kernel Team
  - Linaro

A look into the future

- Next higher-speed devices are likely to suffer from the same latency problems, or even worse
  - Mainly because they will have multiple and deeper internal queues
    - To achieve a high throughput internal queues must be kept constantly filled with high-enough number of I/O requests
  - Details on demand
- BFQ does have limitations
  - Devised for blk and older, slower devices
  - Not designed for extreme parallelism and minimal possible overhead
  - Not designed for deep-queue devices

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Support for blk-mq in MMC

- The BLOCK maintainers consider the old blk layer to be in maintenance phase
- Necessary to make BFQ available for MMC
- Single hardware queue will in the future just be treated as a special case of a multiqueue with, yeah, one queue
- A patch set transforms MMC to use blk-mq
- Huge refactorings necessary to asynchronously send and receive ACKs on block requests
- A huge refactoring patch set exists with very minor or no performance impact
- Target to merge for v4.12

Linaro
connect
Budapest 2017

ENGINEERS
AND DEVICES
WORKING
TOGETHER

CMDQ support for eMMC

- Command Queuing added in the eMMC 5.1 spec.
- Collaboration! Adrian Hunter (Intel) and Ritesh Harjani (Codeaurora). Excellent work!
- Latest series, Software CMDQ and HW CMDQ support!
- The series has matured - bugs are fixed!

Next steps:

- Provide fresh performance numbers.
- Convert to blk-mq. Before or after?

Thank You

#BUD17

For further information: www.linaro.org

BUD17 keynotes and videos on: connect.linaro.org